ENCICLOPEDIA PRACTICA DE LA

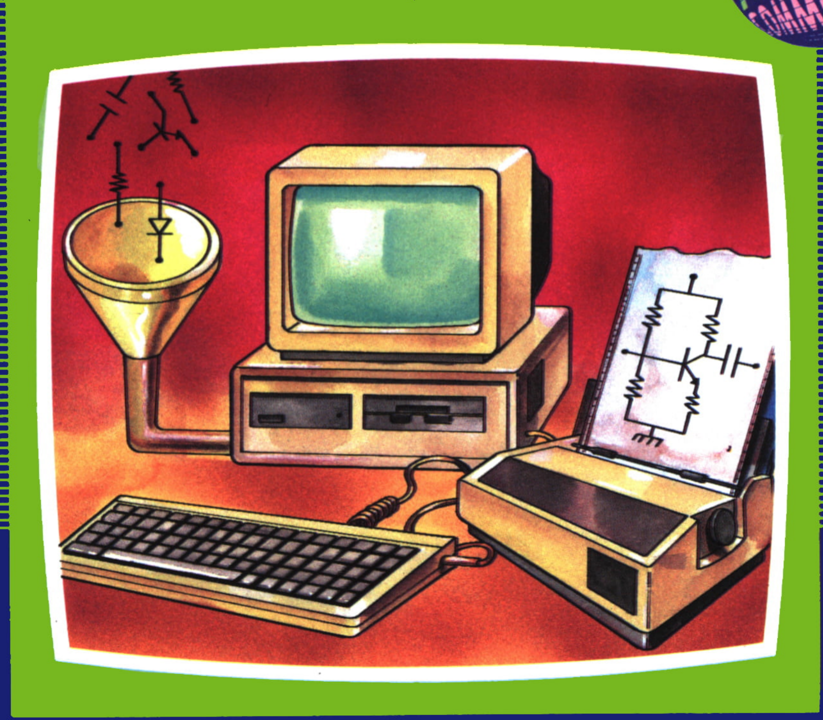
INFORMATICA APLICADA

14

Cómo simular circuitos electrónicos en el ordenador

Antonio Enrique Martínez

SINCLAIR AMSTRAD COMMODORE IBM



EDICIONES SIGLO CULTURAL

ENCICLOPEDIA PRACTICA DE LA

INFORMATICA APLICADA

14

Cómo simular circuitos electrónicos en el ordenador

EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S.A.

Director-editor:
RICARDO ESPAÑOL CRESPO.

Gerente:
ANTONIO G. CUERPO.

Directora de producción:
MARIA LUISA SUAREZ PEREZ.

Directores de la colección:
MANUEL ALFONSECA, Doctor Ingeniero de Telecomunicación
y Licenciado en Informática
JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño y maquetación:
BRAVO-LOFISH.

Dibujos:
JOSE OCHOA Y ANTONIO PERERA.

Tomo XIV. **Cómo simular circuitos electrónicos en el ordenador**
ANTONIO ENRIQUE MARTINEZ, Ingeniero de Telecomunicación.

Ediciones Siglo Cultural, S.A.

Dirección, redacción y administración:
Sor Angela de la Cruz, 24-7.º G. Teléf. 279 40 36. 28020 Madrid.

Publicidad:
Gofar Publicidad, S.A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:
COEDIS, S.A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.
Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:
CADE, S.R.L. Pasaje Sud América. 1532. Teléf.: 21 24 64.
Buenos Aires - 1.290. Argentina.

ISBN del tomo: 84-7688-046-4
ISBN de la obra: 84-7688-018-9.

Fotocomposición:
ARTECOMP, S.A. Albarracín, 50. 28037 Madrid.

Imprime:
MATEU CROMO. Pinto (Madrid).

Depósito legal: M-42.838-1986
Printed in Spain - Impreso en España.

Suscripciones y números atrasados:
Ediciones Siglo Cultural, S.A.
Sor Angela de la Cruz, 24-7.° G. Teléf. 279 40 36. 28020 Madrid

Octubre, 1986.
P.V.P. Canarias: 365,-

INDICE

Los programas que aparecen en este libro funcionan en los ordenadores:

IBM–PC, XT, AT y compatibles.
AMSTRAD–464, 664, 6128, 1512.
SINCLAIR–SPECTRUM 48 K, 128 K, PLUS, PLUS 2.
MSX–Todos los modelos.
COMMODORE–CBM 64 y CBM 128.

INTRODUCCION 1



ESDE que el hombre diseñó y construyó el primer ordenador siempre ha tenido el empeño de extender su uso a todas las áreas posibles de la vida humana. Pensemos en los primeros ordenadores, aun familiares para todos por las imágenes que de ellos hemos recibido a través de los medios de comunicación y espectáculos: nadie se quedaba con la idea de la máquina de calcular que estaba encerrada en un edificio y que suministraba soluciones a las complicadas ecuaciones con las que los matemáticos la alimentaban a través de cientos de tarjetas perforadas. La imagen del ordenador era la del dispositivo que controlaba lo que le rodeaba, que sabía todo sobre todo y sobre todos nosotros, que era capaz de responder a cualquier pregunta que se le formulara. Y esta imagen del ordenador alimentó nuestra mente y la de cientos de escritores de ciencia ficción que se dedicaron a elucubrar sobre el tema.

Literatura aparte, esta es la idea fundamental que debemos destacar al comenzar este libro. El ordenador es una máquina de propósito general, capaz de resolver problemas de la más diversa índole al usuario sin más que utilizar el programa conveniente y los circuitos de control adecuados. Hoy en día tenemos ejemplos claros en la mayoría de los aparatos que nos rodean: el ascensor automático que recuerda todos los pisos en los que debe parar, el horno que nos dice el tiempo de cocinado sin más que suministrarle el alimento a preparar y su peso, la máquina de coser con la que la abuelita hubiera soñado, que cose como ni a la más hábil costurera se le hubiera ocurrido hacerlo...

En el campo de la ingeniería, el ordenador se supo aprovechar desde sus comienzos. Incluso cuando era demasiado voluminoso y demasiado caro para poder introducirlo en una lavadora, no lo era para las grandes empresas, para las universidades y centros de investigación. El técnico contaba con una nueva herramienta de trabajo. Todos los cálculos tedio-

sos, repetitivos que ocupaban la mayor parte de su tiempo podía describirlos en función de simples instrucciones. El ordenador las podía ejecutar tantas veces como él quisiera sobre distintos datos. Así se obtenían las soluciones que luego se podrían transformar en motores, alas de un avión, carenados o radioenlaces.

Quizá movido a reflexionar por el mayor tiempo que el ordenador le permitía emplear en su labor creativa o por cualquier otra razón, el hecho es que cada vez el ingeniero le fue pidiendo más a su ordenador. Poco a poco lo fue programando para tareas más complejas. En especial, para las encaminadas a resolver una de las grandes preguntas que se plantea siempre el constructor de cualquier tipo de aparato: «¿Qué pasaría si...?» Así nacieron los programas de simulación.

¿Qué es simular? Según la Real Academia de la Lengua Española, simular es «Representar una cosa, fingiendo o imitando lo que no es». Entrando más en detalle en el tema que nos ocupa, podemos seguir la definición que Shannon dio en 1975: «Simulación es el proceso de diseñar un modelo de un sistema real y realizar experimentos con este modelo con el propósito de comprender el funcionamiento del sistema o de evaluar diferentes estrategias para su operación.»

Gracias a estas técnicas el ordenador nos puede mostrar lo arrugado que podría quedar nuestro flamante diseño de un avión-cohete si se ve atacado por una lluvia de meteoritos, o lo que sería de nuestro amplificador estereofónico si lo enchufáramos a una red de 380 voltios en vez de nuestros habituales 220.

La simulación tiene una importancia fundamental para el diseño en ingeniería. El diseño es el proceso por el cual, partiendo de unos requerimientos fundamentales o especificaciones, se obtiene un producto final que los satisface. Esto es general para todas las áreas de la ingeniería, y en particular para la ingeniería electrónica.

En el caso de la ingeniería electrónica, intentar abordar este diseño o síntesis de un circuito directamente es casi imposible en cuanto su complejidad sea mediana. Las ecuaciones a resolver nos llevan a unos sistemas complicadísimos en los que prácticamente nunca hallaremos una única solución para nuestras incógnitas (valores de componentes), aun suponiendo que pudiéramos llegar a ellas. Salvo en casos muy concretos, como son el diseño de filtros, no disponemos de métodos que permitan, a partir de unas especificaciones funcionales, llegar a la realización óptima (ni tan siquiera aproximada) del circuito que las cumple.

Es por esto que el diseño electrónico siempre se ha basado en métodos de ensayo/error/repetición. En la figura 1 se puede ver un sencillo organigrama que muestra el proceso normal por el que se construye un circuito de cualquier clase. Partiendo de unas especificaciones externas, dadas por el resto del equipo en el que debe ir integrado el circuito, por nor-

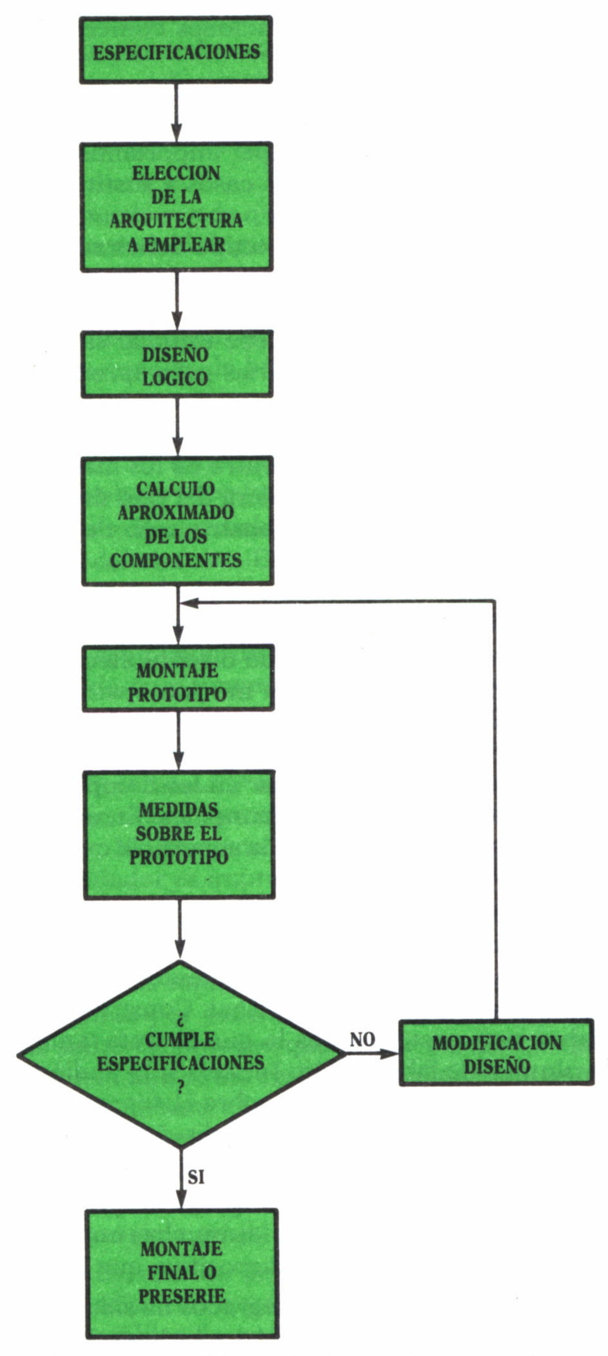


Fig. 1. Metodología de diseño de un circuito.

mas internacionales o por las especificaciones comerciales del aparato a fabricar, se fija una arquitectura.

Para esta arquitectura se hace un cálculo aproximado de los valores de los componentes a utilizar. Aquí, además de la teoría de circuitos, de las matemáticas, siempre ha jugado un papel importantísimo la experiencia del diseñador, llegando esta en algunos casos a sustituir totalmente a las otras.

El siguiente paso es construir un prototipo del circuito, fácilmente modificable. Sobre él se hacen las medidas necesarias para comprobar si cumple o no las especificaciones de partida.

En caso de que no las cumpla, se debe estudiar la razón e introducir en el diseño las modificaciones necesarias para aproximarse a lo que se busca.

Este proceso de medida/alteración se repetirá tantas veces como sea necesario hasta que nos encontremos dentro de los márgenes establecidos de funcionamiento, cuya rigidez dependerá del tipo de aplicación. El último prototipo servirá de base para la construcción de una preserie o del equipo final.

Por este procedimiento se han desarrollado durante mucho tiempo los circuitos convencionales (placas de un televisor, receptores de radio...).

Pero consideremos ahora otro tipo de diseños electrónicos: el de esos diminutos artefactos de color negro con muchas «patitas» por cada lado y que los expertos llaman «chips». Un chip o circuito integrado no es más que un circuito electrónico convencional miniaturizado hasta unos extremos que jamás hubiesen podido imaginar en los tiempos en que la válvula de vacío era el «best-seller» de los componentes. En un chip puede haber miles de transistores, ¡y más del 50% de la superficie que vemos es sólo envoltorio!

Pensemos, pues, en el proceso de diseño de uno de estos circuitos. Sus características eléctricas le vienen dadas en gran parte por su tamaño. No se puede construir un prototipo con componentes convencionales y esperar que responda igual que el producto final. Construir un prototipo integrado es una labor muy costosa, pues lo que abarata tanto estos dispositivos es el gran volumen de fabricación. Incluso si lo pudiéramos construir, no sería posible realizar modificaciones sobre él, teniendo que realizar uno nuevo en caso de que existan errores en el diseño.

La única salida posible en este caso es la simulación. Una simulación que considere todas las características especiales que inciden sobre este tipo de circuitos y nos ayude a respetar las muchas consideraciones de diseño imprescindibles para realizar un dispositivo que funcione.

Esta complejidad de los circuitos integrados ha sido el gran motor para el desarrollo de programas de simulación y la investigación sobre cómo utilizar el ordenador en el trabajo de concepción.

Para este fin es para lo que se han desarrollado los programas llamados de diseño asistido por ordenador / fabricación asistida por ordenador (más conocidos por sus siglas inglesas CAD / CAM : Computer Aided Design / Computer Aided Manufacturing). Prácticamente todas las áreas de la ingeniería en muy diversos campos en cada una de ellas poseen hoy programas de este tipo.

En la figura 2 se puede ver de qué manera influye un ordenador en este proceso de diseño. La metodología es prácticamente la misma, pero tenemos las siguientes ayudas:

- La arquitectura a emplear muchas veces se reduce a elegir entre los distintos bloques funcionales de los que el ordenador ya posee en su memoria de almacenamiento masivo. Por ejemplo, un bloque funcional puede ser «fuente de alimentación de 5 voltios y 10 amperios», que, en virtud de su uso en muy diversas situaciones, se puede haber construido con anterioridad y emplear ahora directamente o con cambios muy pequeños.
- Entrada de datos: Una vez elegida la arquitectura a emplear hay que construir los planos del circuito. El ordenador nos auxiliará en esta labor, permitiéndonos dibujar, de forma sencilla, los esquemas funcionales de los distintos elementos y ofreciéndonos procedimientos de edición del circuito: conexionado, repetición de bloques, borrado de conexiones, inserción y eliminación de componentes... Todos estos datos los almacenará para utilizarlos posteriormente en las distintas partes del proceso.
- Cálculo de componentes: No siempre es sencillo establecer cuál será su valor «a priori». El ordenador nos puede auxiliar con diversos métodos de cálculo para obtener una aproximación lo suficientemente buena para comenzar a trabajar. No obstante, es este un paso en el que la intervención directa del diseñador es imprescindible, pues el sentido común, en la mayoría de los casos, suele ser la mejor herramienta y la más cómoda de utilizar.
- Simulación: Esta es una de las principales ventajas que nos proporciona el diseño asistido por ordenador. En lugar de montar el circuito y realizar pruebas sobre él, proceso que puede resultar largo y costoso, el mismo ordenador nos dirá cómo responderá el circuito ante los estímulos externos que le fijemos. Para ello necesita saber cómo se comporta cada elemento básico. Esto lo conoce gracias a la biblioteca de modelos de dispositivos de que dispone. El modelo que emplee, en especial para componentes activos como transistores y diodos, es vital a la hora de obtener una simulación lo más próxima posible al caso real.
- El proceso de corrección del circuito es ahora inmediato. Basta con acudir de nuevo al editor de entrada de componentes para alterar un valor de resistencia, cambiar de transistor o añadir un nuevo condensador de desacoplo.

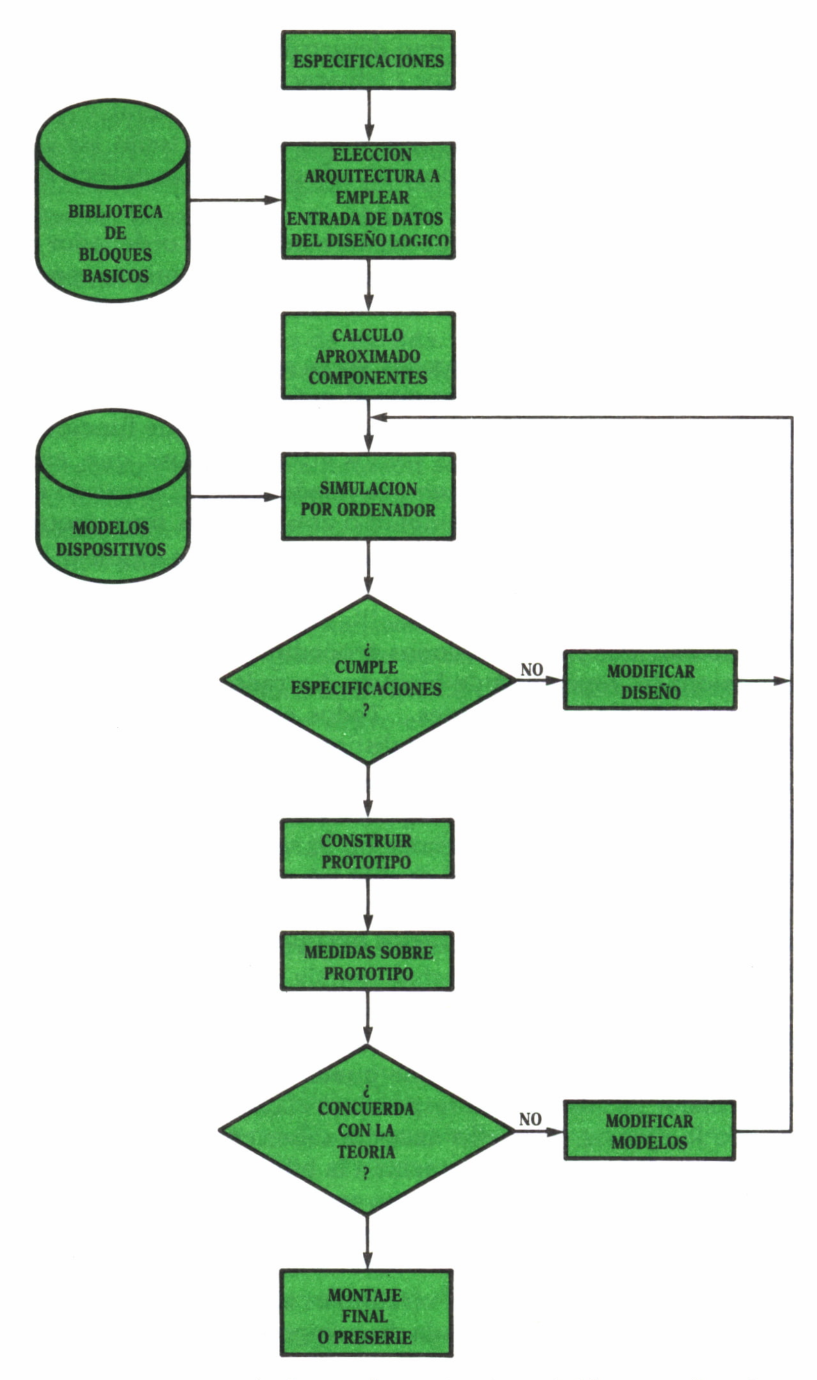


Fig. 2. Proceso de diseño de un circuito asistido por ordenador.

• Una vez que el análisis del circuito con el ordenador nos indica que las especificaciones se van a cumplir, debemos construir un prototipo para validar los cálculos. En caso de que el prototipo no responda a lo esperado, debemos reconsiderar los modelos que se han empleado para la simulación y sustituirlos por otros más reales o más cercanos al caso.

• Tras finalizar el diseño y validarlo con un prototipo, en el método tradicional quedaba una de las tareas más tediosas: el paso del diseño lógico a un diseño físico definitivo. La realización de una tarjeta de circuito impreso es siempre una de las tareas más laboriosas en cualquier proceso.

Hoy, partiendo del diseño lógico, que ya tenemos almacenado en nuestro ordenador, podemos obtener el circuito impreso deseado de una forma casi automática. El ordenador nos auxiliará en las tareas de:

• Colocación de los dispositivos de forma óptima.

• Conexionado minimizando diversos parámetros: longitudes de pistas, tramos paralelos de señales, pasos de cara, problemas de ruido.

• Generación de las máscaras para la realización física del circuito.

En todo el proceso de evolución del trabajo de diseño podemos apreciar una característica fundamental, que muchas veces se nos olvida considerar a la hora de echar a volar nuestra imaginación: la mente del hombre está siempre por delante del trabajo que realiza el ordenador. No es el ordenador el que construye un yate, diseña un televisor o establece qué forma ha de tener el nuevo modelo de deportivo. El diseñador lleva a la práctica sus ideas utilizando distintas herramientas, y una de ellas y muy importante en la actualidad es el ordenador.

En cualquier caso, conviene resaltar que normalmente nunca se habla de «diseño de circuitos por ordenador» simplemente, sino de «diseño de circuitos asistido por ordenador». En la palabra «asistido» se encierra gran parte del secreto de los ordenadores que, sólo en contadas ocasiones, escapa del círculo de usuarios técnicos y se difunde en la sociedad. El diseño en ingeniería, no sólo electrónica, como ya hemos visto, normalmente tiene una cantidad de grados de libertad tan grande que el plantear un problema general y hallar el método de programarlo en un ordenador es poco más o menos que imposible. Esto sólo se realiza en casos muy concretos y para pequeños problemas de los que se conoce muy bien la solución óptima y se puede delegar en el ordenador todas las tareas. En el resto de los casos, el diseñador se apoyará en el ordenador para ir realizando su trabajo. Este puede ayudarle en tareas de la más diversa índole:

• Formas o módulos básicos que puede ensamblar, como un rompecabezas, para componer su producto. Para ello se ayudará de su gran capacidad de almacenamiento para mantener una biblioteca de módulos.

• Normas de conexionado de módulos: puede detectar irregularidades en los conexionados que darán lugar a un mal funcionamiento.

• Dibujo automático de los planos y guías de montaje, con la gran ventaja de poder regenerarlos de nuevo tras un cambio sin producir el doble de trabajo de delineante.

• Simulación del producto en las condiciones que se establezca, para evaluar si su comportamiento responde a las perspectivas para las que se diseñó.

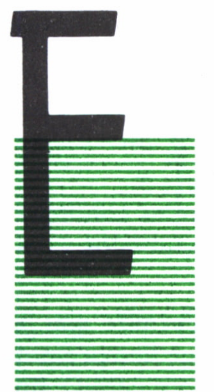
Así se podrían seguir dando ejemplos de situaciones en las que un ordenador puede asistir a la hora de diseñar cualquier cosa.

El presente libro va a intentar introducirnos en las técnicas y herramientas destinadas a realizar las tareas de análisis de un circuito de modo automático, y a aprovechar los conocimientos que se obtienen del análisis para la ayuda en la síntesis.

A lo largo de él se introducirán programas que intenten llevar a la práctica las ideas y los algoritmos matemáticos expuestos. El fin fundamental que persiguen es didáctico y, por tanto, en ellos se ha sacrificado en muchos casos la potencia o la rapidez de ejecución en favor de la claridad. Se pretende que el lector pueda extraer de ellos las ideas suficientes para atacar su propio problema de una forma más eficiente.

Precisamente por esta razón es por lo que los programas se han desarrollado en lenguaje Pascal, fruto de los esfuerzos de la programación estructurada, y cuya validez como medio de transmisión de ideas en el campo de la programación está más que probada y avalada por su extendido uso, no sólo como lenguaje en sí, sino también como metalenguaje en descripción de algoritmos. Para facilitar la labor de comprensión al lector que no esté muy familiarizado con él se ha tratado de utilizar sólo las instrucciones más sencillas del repertorio disponible en cualquier buen compilador. De este modo se facilita también la tarea de adaptación para quien desee construir sus futuros programas de diseño en otros lenguajes.

FUNDAMENTOS DEL DISEÑO DE CIRCUITO ASISTIDO POR ORDENADOR 2



ESTE capítulo trata de uno de los temas que suele presentar menos interés al aficionado, pero cuyo conocimiento es imprescindible tanto para el profesional como para el que se quiere tomar un poco más en serio el pasatiempo de los ratos de ocio. Se trata, naturalmente, de las matemáticas a emplear para poder representar un circuito de cara al ordenador y realizar cálculos sobre él.

En todo este capítulo se supone que el lector posee unos conocimientos matemáticos sobre lo que es la derivación, los números complejos y las funciones trigonométricas, que no se desarrollan por quedar fuera del tema y la extensión del presente libro. De todas formas, no se preocupe si éste no es su caso. Lea usted el primer apartado, en el que se dará una descripción informal de los componentes empleados para definir una red, y pase al final del capítulo, en donde encontrará, ampliamente comentado, un programa que le permitirá pasar de un circuito electrónico a las ecuaciones que lo representan.



DEFINICIONES FUNDAMENTALES

Una red eléctrica es un conjunto de componentes tales como resistencias, condensadores, bobinas, etc., ligados entre sí mediante conductores.

Una red es lineal cuando está constituida por elementos lineales y bilaterales, que son en los que las ecuaciones que los representan son independientes de la dirección de la corriente o de lo grande que sea su valor.

Elementos lineales típicos son las resistencias, bobinas y condensadores (no electrolíticos), ya que en ellos no influye la posición en que se conecten y se siguen comportando igual por muy grande que sea la corriente que los atraviese, siempre, por supuesto, ¡que no se lleguen a quemar!

Aunque más adelante se definirán formalmente los citados elementos, vamos a ver aquí de una forma sencilla sus características fundamentales:

- Una resistencia o resistor es un componente eléctrico que se «opone» al paso de una corriente por él. Esta oposición es tanto mayor cuanto mayor sea su valor, que medimos con una unidad denominada ohmio. Una característica fundamental de estos componentes es que transforman parte de la energía eléctrica que conducen en calor.
- La bobina es un componente constituido por un conductor enrollado sobre un núcleo, normalmente cilíndrico. Su propiedad fundamental es la de almacenar energía en forma de campo magnético.
- Un condensador es un componente constituido por dos conductores próximos entre sí y separados por un elemento aislante o dieléctrico. Al igual que la bobina, son capaces de almacenar energía, pero en este caso se realiza en forma de campo eléctrico.

Estos circuitos los podemos ver representados en la figura 3.

Elementos no lineales son aquéllos en los que no se dan las condiciones establecidas para los lineales. Ejemplos clásicos son los diodos y los transistores.

- El diodo es un componente de dos terminales que permite que la corriente circule por él siempre que entre por el terminal que llamaremos *ánodo* y, por tanto, salga por su otro terminal, que llamaremos *cátodo*. En caso contrario, no deja pasar la corriente eléctrica.
- Un transistor es un elemento de tres terminales. Hay distintos tipos de ellos, pero todos coinciden en que la corriente que circula entre dos de sus terminales se puede controlar con el tercer terminal:
  - En el caso de los llamados *transistores bipolares,* el control se realiza mediante la corriente que se hace pasar por el tercer terminal. Los terminales controlados se denominan *colector* (C) y *emisor* (E), y el terminal de control, *base* (B). Según los sentidos de las corrientes que lo atraviesan, hay dos tipos de transistores bipolares:
    - NPN: en ellos la corriente de control entra en la base, y la controlada circula de colector a emisor.
    - PNP: exactamente al contrario: la corriente sale de la base y la controlada circula de emisor a colector.
  - En los llamados transistores de efecto de campo (FET), es la tensión aplicada a uno de sus terminales, que llamamos *puerta,* la que controla la corriente entre los terminales llamados *drenador* (D) y *fuente* (S).

Los circuitos a los que nos referiremos de momento serán lineales. Más adelante estudiaremos lo que se debe hacer para analizar los componentes no lineales.

Otros componentes básicos de una red son los generadores, elementos que introducen la señal en nuestro circuito. Los podemos dividir en dos grandes grupos:

- Generadores independientes, cuyo valor de tensión no varía con el valor de cualquier otro parámetro de la red.
- Generadores controlados, en caso contrario.

En el siguiente apartado se estudian con más detalle dichos componentes.

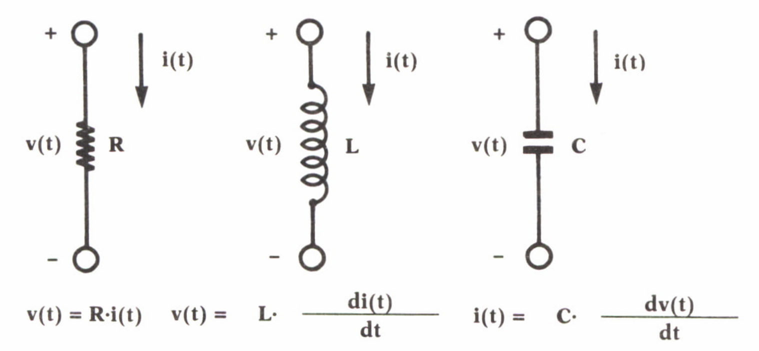


Fig. 3. Elementos básicos de una red y ecuaciones que los representan.



RELACIONES CORRIENTE-TENSION

Tenemos como fin representar una red eléctrica de forma matemática. Debemos comenzar, por tanto, por estudiar la forma en que podemos representar los elementos que la componen. Veamos la figura 3. En ella se ven los elementos más sencillos que podemos considerar. Por ellos circula una corriente que varía con el tiempo, que llamaremos *i(t)* y entre sus extremos hay una diferencia de tensión que denominaremos v(t). Para definir su comportamiento eléctrico es necesario dar una ecuación que relacione estas dos magnitudes, denominada (¿se le ocurriría a usted un nombre mejor?) relación corriente-tensión. Así tenemos:

- Resistencias: Su relación corriente-tensión viene dada por la Ley de Ohm en su forma más simple: la tensión en bornas de una resistencia es igual al producto de la corriente que por ella circula por una constante:

$$v(t) = R\, i(t)$$

El valor de la constante es lo que llamamos *resistencia*, y se mide en ohmios.

• Bobinas: En este caso la relación no es tan simple. La tensión en bornas de una bobina es igual al producto de una constante por la derivada de la corriente que por ella pasa respecto al tiempo:

$$v(t) = \frac{di(t)}{dt}$$

La constante *L* es lo que se denomina valor inductivo de una bobina o autoinducción, y su unidad de medida es el henrio.

• Condensadores: En estos casos la relación es inversa a la de las bobinas: la corriente que atraviesa un condensador es igual al producto de otra nueva constante por la derivada de la tensión en sus extremos respecto del tiempo:

$$i(t) = C\,\frac{dv(t)}{dt}$$

En este caso la constante se denomina capacidad del condensador, y su unidad de medida es el faradio.

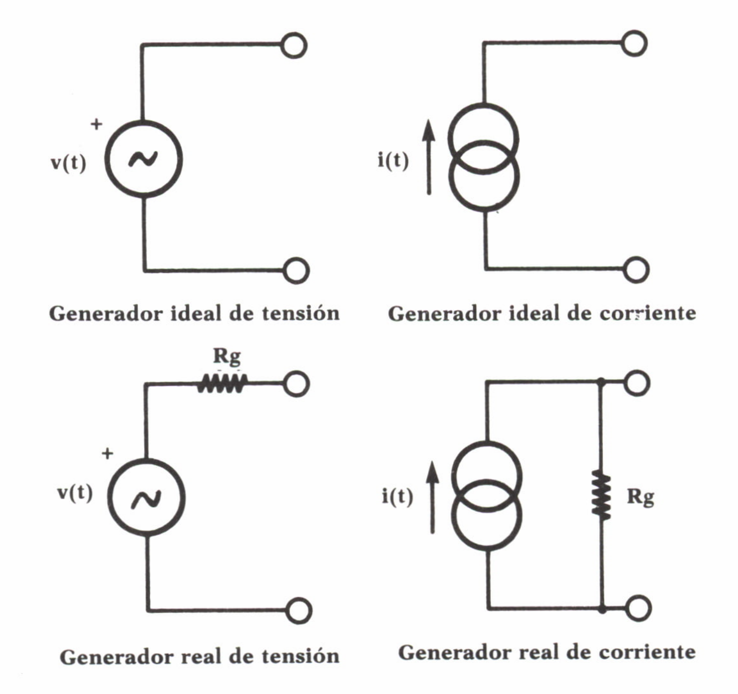


Fig. 4. Generadores de excitación de la red.

Los elementos que actúan como excitadores de nuestra red reciben el nombre de generadores. En un amplificador el generador sería el disposi-

tivo que colocamos a su entrada: un micrófono o la salida del tocadiscos, por ejemplo. En la figura 4 se puede ver su representación sobre el papel.

Un generador ideal es el que es capaz de suministrar su tensión o su corriente, según el tipo de que se trate, independientemente de las características y valores de los componentes que se coloquen a su salida. Estos generadores, como es lógico, no existen en la realidad. Sin embargo se estudian porque un generador real se puede considerar como un generador ideal con una determinada resistencia interna, que limita su funcionamiento y lo hace dependiente del exterior.

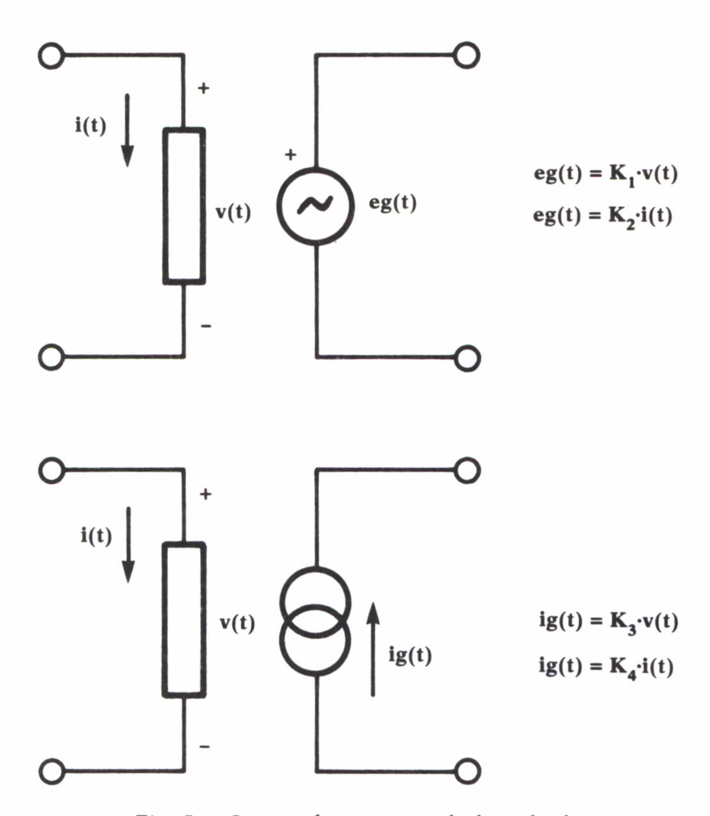


Fig. 5. Generadores controlados ideales.

Otros de los componentes normalmente empleados para representar circuitos electrónicos, sobre todo para el modelado de dispositivos complejos mediante elementos simples, son los generadores controlados. En la figura 5 se puede ver su diagrama y sus relaciones corriente-tensión. Como se puede apreciar, el valor de tensión o corriente que otorgan no es, como en el caso anterior, constante, sino que depende del valor de otra magnitud en cualquier parte del circuito.

También en este caso podríamos hablar de generadores controlados ideales y reales, al igual que para los generadores independientes. Sus representaciones se ven en la siguiente figura:

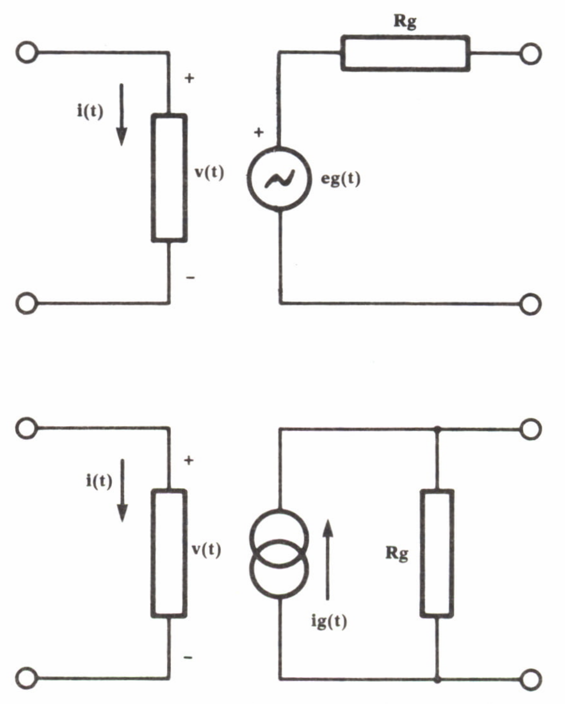


Fig. 6. Generadores controlados reales.

Para terminar, veamos las ecuaciones que rigen el comportamiento de dos inductancias con acoplamiento mutuo o transformador.

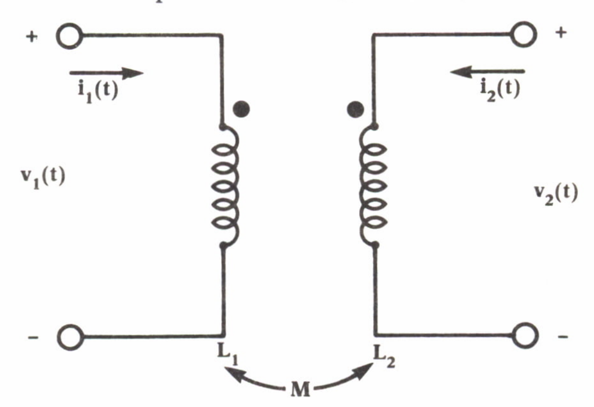


Fig. 7. Representación esquemática de un transformador.

Los puntos junto a las bobinas indican por cuál de sus extremos deben entrar las corrientes al transformador para que los flujos magnéticos creados por ambos se sumen.

Las relaciones corriente-tensión en este caso son de la forma siguiente:

$$v1(t) = L1 \frac{di1(t)}{dt} + M \frac{di2(t)}{dt}$$

$$v2(t) = L2 \frac{di2(t)}{dt} + M \frac{di1(t)}{dt}$$

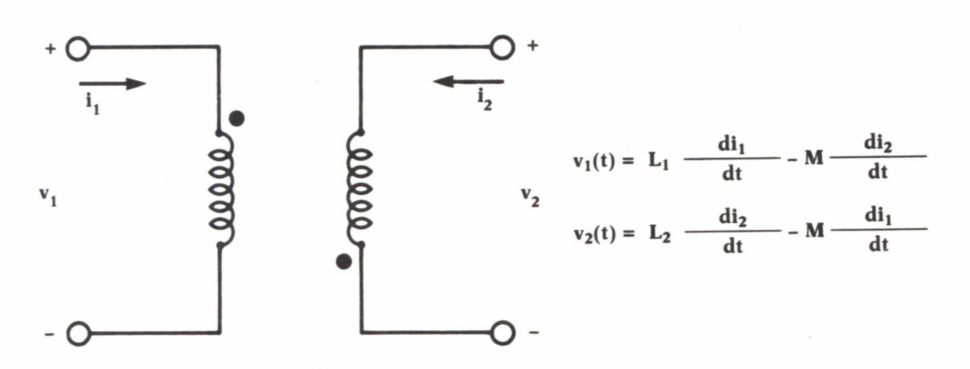
Se pueden apreciar dos términos que relacionan ahora la corriente con la tensión en cada bobina:

- El primero es el debido a su autoinducción, igual al que se tiene en caso de no existir acoplamiento.
- El segundo es debido a la corriente por la bobina acoplada, y tiene la misma forma que el primero. El coeficiente *M* se denomina *coeficiente de acoplamiento mutuo,* y depende de la geometría del transformador.

Este término representa la tensión que induce en un devanado la corriente que circula por el contrario. Esta tensión cambia de polaridad al cambiar el sentido de la corriente en el devanado inductor, lo que es importante a la hora de escribir las ecuaciones. Podemos establecer como norma general:

Una corriente que entra por el extremo de la bobina marcado con el punto inducirá una tensión positiva en el extremo de la otra bobina también marcado con el punto, y viceversa.

La siguiente figura nos muestran la diferencia de las ecuaciones en caso de que las corrientes por los devanados sean de distintos sentidos:



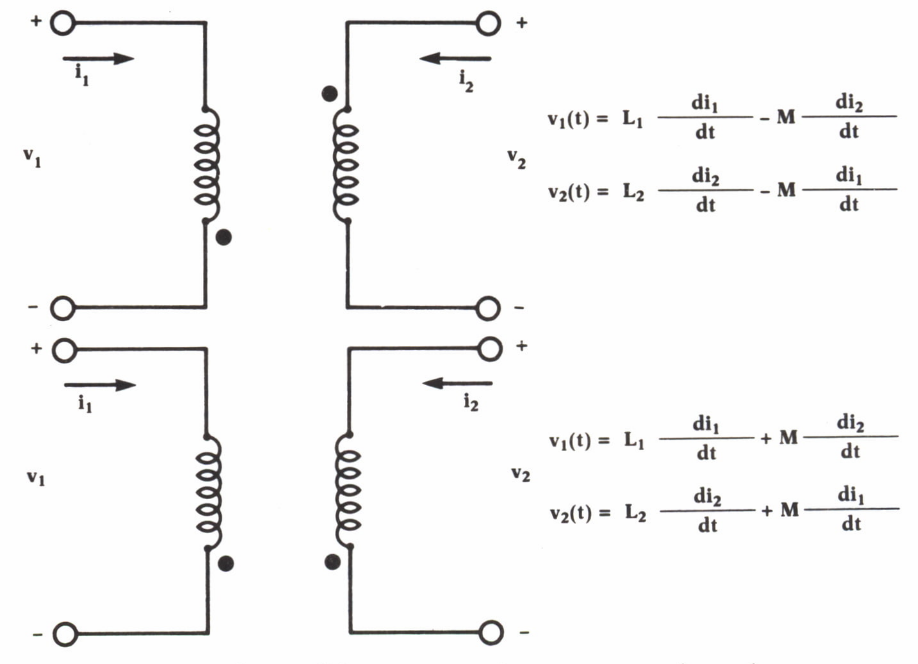


Fig. 8. Flujos aditivos y sustractivos en un transformador.



IMPEDANCIAS

Como vemos, la ligadura entre las magnitudes que nos interesan no es simple. Aparecen derivadas. El análisis de un circuito por este camino nos llevaría a la resolución de un sistema de ecuaciones diferenciales, lo cual no es probablemente lo que más nos apetecía.

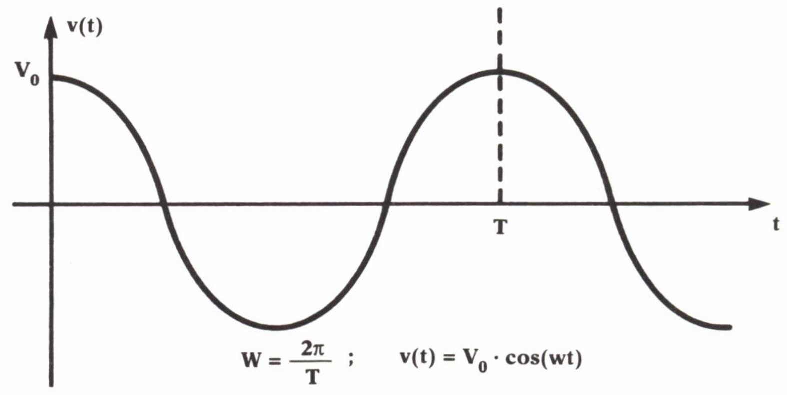


Fig. 9. Representación de una función sinusoidal en el tiempo.

Para evitar este conflicto demos un rodeo. Vamos a considerar que los generadores que atacan nuestro circuito siguen una variación de tipo sinusoidal (ver figura 9). En este caso, y supuesto que ha pasado mucho tiempo desde que se encendió el circuito (a esto se le llama *régimen permanente*), todas las corrientes y tensiones que por él circulen son también sinusoidales.

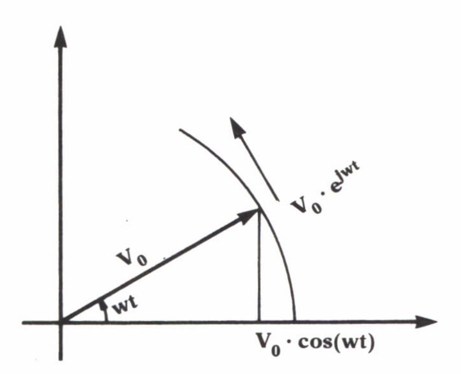


Fig. 10. Representación de una función sinusoidal por un vector.

Las señales sinusoidales se pueden representar por la proyección de un vector giratorio sobre un eje horizontal (ver figura 10). El vector lo podemos representar como un número complejo en forma módulo-argumento con la siguiente notación:

$$V\ e^{j\omega t}$$

expresión en la cual *j* es la unidad imaginaria ($\sqrt{-1}$).

Esta forma de representar un número complejo es equivalente a otras que probablemente ya conocerá el lector, como pueden ser:

- Forma binomial: a + j * b = cos(ωt) + j * sen(ωt)
- Forma módulo - argumento: V ωt

Conviene destacar también que es corriente en toda la teoría de circuitos representar la unidad imaginaria con la letra j en lugar de con la letra i, como es habitual en matemáticas, debido a la confusión que originaría si se hiciera de esta última forma, por ser la letra i con la que normalmente se designan las intensidades de corriente.

Según esta expresión, la señal sinusoidal representada en la figura 9 sería la parte real del vector así considerado:

$$v(t) = \text{Re}\ [V\ e^{j\omega t}] = Vo\ \cos\ \omega t$$

Normalmente en los cálculos no se toma esta parte real más que al final de las operaciones, y se trabaja durante todo el problema con las magnitudes complejas.

Si aplicamos ahora esta formulación a las relaciones corriente-tensión que se han visto para los componentes básicos, obtendríamos:

$$V\, e^{j\omega t} = R\, I\, e^{j\omega t} \qquad \text{para las resistencias}$$

$$V\, e^{j\omega t} = L \frac{d\,(I\, e^{j\omega t})}{dt} = j\omega L\, I\, e^{j\omega t} \qquad \text{para las bobinas}$$

$$I\, e^{j\omega t} = C \frac{d\,(V\, e^{j\omega t})}{dt} = j\omega C\, V\, e^{j\omega t} \qquad \text{para los condensadores}$$

Si despejamos de estas ecuaciones el cociente entre los valores máximos de tensión y corriente obtenemos:

$$\frac{V}{I} = R \quad = Zr \qquad \text{en una resistencia}$$

$$\frac{V}{I} = j\omega L \quad = Zl \qquad \text{en una bobina}$$

$$\frac{V}{I} = \frac{1}{j\omega t} = Zc \qquad \text{en un condensador}$$

que son las expresiones de las *impedancias* de la resistencia, bobina y condensador, respectivamente. Los cálculos se realizan para valores máximos de excitación, con lo cual no se maneja el término de la exponencial. En virtud de esto, las expresiones anteriores son las *relaciones corriente tensión* de los elementos básicos *en régimen permanente sinusoidal.*

Es también interesante conocer que a los inversos de dichas cantidades se les denomina *admitancias.* Así tendremos:

$$\frac{1}{Zr} = Yr \qquad \text{Admitancia de la resistencia.}$$

$$\frac{1}{Z1} = Y1 \qquad \text{Admitancia de la bobina.}$$

$$\frac{1}{Zc} = Yc \qquad \text{Admitancia del condensador.}$$

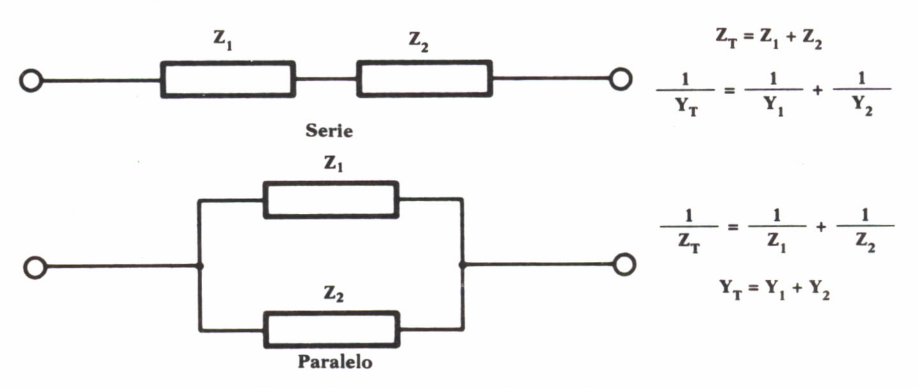


Fig. 11. Asociación de impedancias.

Para las impedancias rigen las mismas normas de asociación que para las resistencias, esto es (ver figura 11):

- Dos impedancias conectadas en serie suman su valor.

- El inverso de la impedancia total de una conexión en paralelo de impedancias es igual a la suma de los inversos de los valores de las impedancias que se conectan.

Fácilmente se ve, sin más que considerar su definición, que con las admitancias estas reglas son a la inversa.

Del mismo modo que transformamos las ecuaciones de los componentes sencillos, podemos transformar las de las inductancias acopladas, obteniendo unas ecuaciones de la forma:

$$V1 = j\omega L1\ I1 + j\omega M\ I2$$

$$V2 = j\omega L2\ I2 + j\omega M\ I1$$

considerando los sentidos de corrientes y tensiones como en la figura 7.



LEMAS DE KIRCHHOFF

Ahora ya se han visto los componentes básicos de una red y las relaciones que los definen. Pasemos a estudiar ahora cómo atacar el problema de su interconexión.

Veamos un ejemplo sencillo de red:

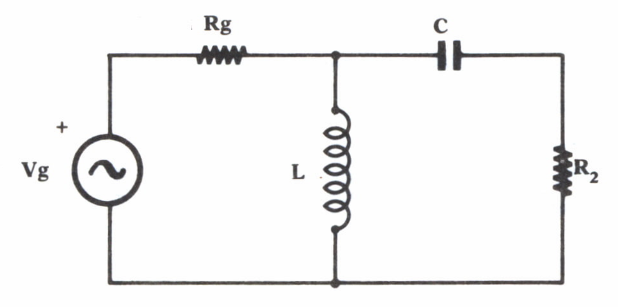


Fig. 12. Ejemplo sencillo de una red lineal.

• Llamaremos *malla* a todo camino cerrado por el que pueda circular una corriente en nuestra red. En la red de la figura son mallas las formadas por los componentes:

- Vg, Rg y L.
- L, C y R2.
- Vg, Rg, C y R2.

Hay que llamar la atención sobre la última malla, normalmente no considerada, pero que tiene carácter de tal.

• Llamaremos *nudo* al punto donde confluyen dos o más componentes. En la red de la figura son nudos los puntos de intersección de los componentes:

- Vg y Rg.
- Rg, L y C.
- C y R2.
- Vg, L y R2.

El último de ellos es un único nudo, aunque parezca «largo», debido a las líneas que unen los extremos aparentes de los componentes.

Las leyes o lemas de Kirchhoff se refieren a estas dos entidades definidas, y dicen así:

1. La suma de las corrientes que entran en un nudo es igual a la suma de las corrientes que salen de él.
2. La suma de las caídas de tensión en una malla es nula.

Es muy importante señalar el sentido de corrientes y tensiones en cada componente a la hora de aplicar estas reglas. Normalmente se consideran las tensiones positivas en el mismo sentido que circula la corriente por el componente.

Este sentido establecido «a priori» es arbitrario. Al resolver el problema, si el sentido tomado era equivocado, el resultado obtenido será negativo.

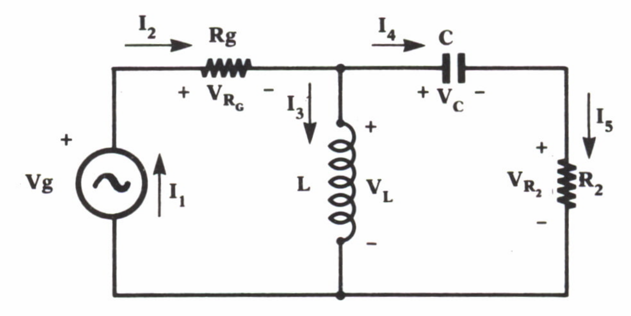


Fig. 13. Asignación de corrientes a una red.

Para nuestro circuito, y considerando los sentidos de corrientes y tensiones de la figura 13 tenemos:

- Ecuaciones de corrientes:

I1 = I2
I2 = I3 + I4
I4 = I5

- Ecuaciones de tensiones:

Vg = Vrg + Vl
Vl = Vc + Vr2

Se han omitido las ecuaciones del nudo **Vg-L-R2** y de la malla **Vg-Rg-C-R2** por ser redundantes con el resto, es decir, a partir de las presentadas se pueden obtener aquéllas sin más que realizar transformaciones algebraicas.

Con las ecuaciones establecidas y las relaciones corriente-tensión de cada componente se resuelve el circuito. En el caso más general, tomaremos las relaciones corriente-tensión en su forma diferencial. La resolución del sistema de ecuaciones diferenciales al que llegaríamos nos daría la solución total del problema. Normalmente el problema no se ataca directamente así, sino que se divide en varios casos particulares que simplifican el estudio.



ANALISIS NODAL

El método de resolución de circuitos más empleado en la mayoría de los programas de análisis por ordenador se deriva de la aplicación de los lemas de Kirchhoff de corrientes. Este es el método conocido como *análisis nodal,* llamado así porque toma como variables del sistema de ecuaciones a resolver las tensiones en todos los nudos que configuran el circuito.

La única restricción necesaria para aplicar este método es que el circuito no contenga fuentes ideales de tensión. En este caso se debe acudir a un análisis nodal modificado, que toma como variables independientes también las corrientes que dichos generadores ideales suministran al circuito.

Esta restricción no es de vital importancia, ya que los generadores ideales de tensión, por supuesto, no existen en la realidad, y en cualquier caso práctico vamos a tener generadores reales.

Una vez establecida esta descripción, estableceremos aquí las normas prácticas para plantear las ecuaciones nodales de un circuito. El lector interesado en el proceso por el cual se obtienen estas reglas de manera teó-

rica lo puede encontrar en el «Análisis nodal de un circuito», en la página 121.

En todo circuito debemos fijar en primer lugar un nudo de dato o de referencia, del cual no plantearemos ecuaciones, ya que si lo hiciéramos no introduciríamos información adicional para la resolución, pues podríamos darnos cuenta de que dicha ecuación se puede obtener en función de las demás. Normalmente se toma el nudo marcado como «tierra» o «masa», por ser además la referencia natural del circuito, pero podríamos tomar cualquier otro.

Para un circuito con *n* nudos (prescindiendo ya del de masa), el análisis nodal nos proporciona *n* ecuaciones con *n* incógnitas, que podemos representar de la siguiente manera:

Y11 . V1 + Y12 . V2 + ... + Y1i . Vi + ... + Y1n . Vn = Ieq1

Y21 . V1 + Y22 . V2 + ... + Y2i . Vi + ... + Y2n . Vn = Ieq2

....

Yi1 . V1 + Yi2 . V2 + ... + Yii . Vi + ... + Yin . Vn = Ieqi

....

Yn1 . V1 + Yn2 . V2 + ... + Yni . Vi + ... + Ynn . Vn = Ieqn

o bien en forma reducida:

[Yn] . [Vn] = [Ieq]

Donde [Yn] es la *matriz de admitancia nodal reducida*, de dimensiones n por n; [Vn] es el vector de tensiones en los *n* nudos del circuito (nuestras incógnitas); e [Ieq] es el vector de generadores de corriente equivalentes que entregan corriente a cada uno de nuestros n nudos.

- Determinación de los términos de la matriz de admitancias [Yn].

Cada uno de los elementos de esta matriz, que denominaremos genéricamente Yij, está constituido por una suma de términos, cada uno de los cuales representa la contribución al circuito de los elementos unidos a los nudos i, j o relacionados con ellos.

En todos los casos se considera que las corrientes circulan partiendo del primer nudo citado en dirección al segundo, y que las tensiones son positivas en el primer nudo mencionado. Esto es: en un generador de corriente entre los nudos k y l, la corriente irá del nudo k al l; si el generador fuese de tensión, el nudo k sería el conectado al punto positivo de dicho generador.

1. Contribución de resistencias, bobinas y condensadores: Un elemento de este tipo, de admitancia Ya, conectado entre los nudos k y l del circuito, da origen a cuatro términos en [Yn]:

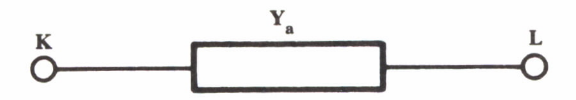


Fig. 14. Admitancias pasivas.

Ykk: Ya Ykl: –Ya

Ylk: –Ya Yll: Ya

2. Generador de corriente controlado por tensión: Supongamos que tenemos un generador de este tipo, de valor Gg, conectado entre los nudos k y l, cuyo valor depende de la tensión existente entre los nudos r y s. Este generador introduce cuatro términos en [Yn], cuyos valores son:

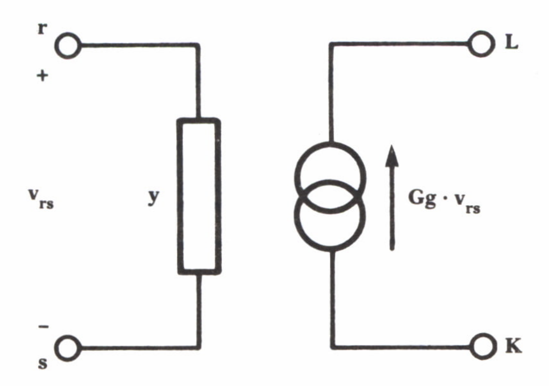


Fig. 15. Generador de corriente controlado por tensión.

Ykr: Gg Yks: –Gg

Ylr: –Gg Yls: Gg

3. Generador de corriente controlado por corriente: Un generador de este tipo, de valor Cg, conectado entre los nudos k y l, cuyo valor depende

de la corriente que recorre una admitancia de valor Yv conectada entre los nudos r y s, introduce cuatro términos en [Yn], de valores:

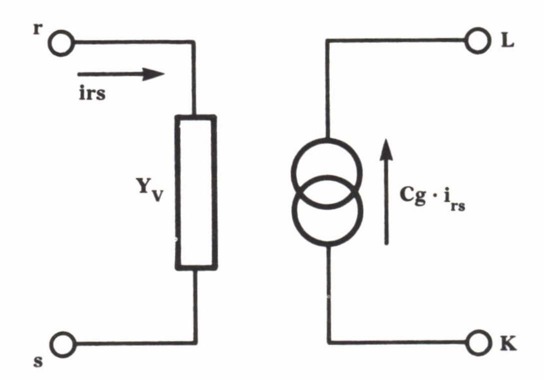


Fig. 16. Generador de corriente controlado por corriente.

Ykr: Cg . Yv Yks: –Cg . Yv
Ylr: –Cg . Yv Yls: Cg . Yv

4. Generador de tensión controlado por corriente: Dicho generador, de valor Dg y con una admitancia serie Yg, conectado entre los nudos k y l, cuyo valor depende de la corriente que recorre una admitancia de valor Yv conectada entre los nudos r y s, introduce cuatro términos en [Yn], de valores:

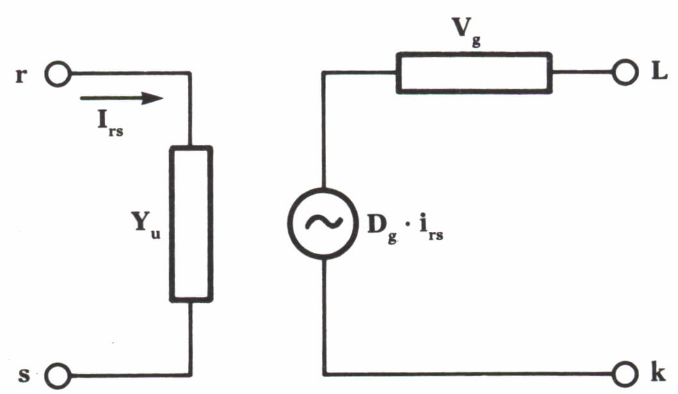


Fig. 17. Generador de tensión controlado por corriente.

Ykr: –Yg . Dv . Yv Yks: Yg . Dv . Yv
Ylr: Yg . Dv . Yv Yls: –Yg . Dv . Yv

5. Generador de tensión controlado por tensión: En este caso, para un valor Bg con una admitancia serie Yg, conectado entre los nudos k y l, con un valor que depende de la corriente que recorre una admitancia de valor Yv conectada entre los nudos r y s, los términos introducidos en [Yn] tienen los siguientes valores:

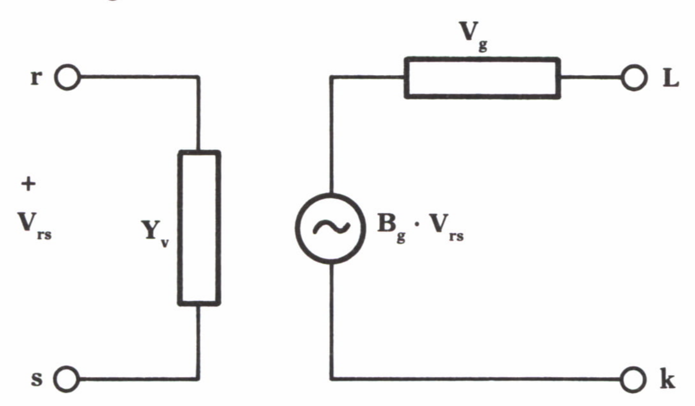


Fig. 18. Generador de tensión controlado por tensión.

Ykr: –Yg . Bv Yks: Yg . Bv
Ylr: Yg . Bv Yls: –Yg . Bv

• Determinación de los términos del vector de generadores de corriente equivalentes [Ieq].

Supondremos en este caso los mismos criterios de signos establecidos para la matriz de admitancias.

1. Generador independiente de corriente, de valor Ig, conectado entre los nudos i y j: Da lugar a dos términos:

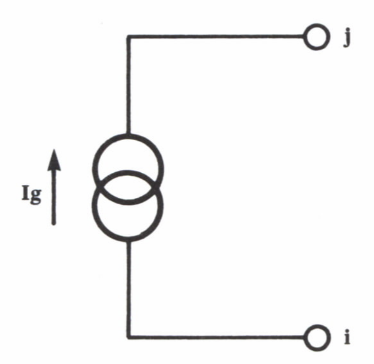


Fig. 19. Generador independiente de corriente.

Ieqi: –Ig
Ieqj: Ig

2. Generador independiente de tensión, de valor Vg, conectado entre los nudos i y j, con admitancia en serie de valor Yg: Da lugar a dos términos:

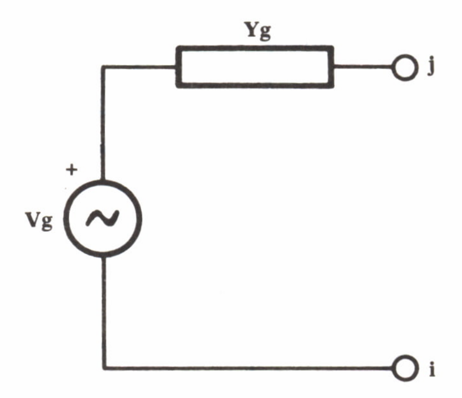


Fig. 20. Generador independiente de tensión.

Ieqi: –Vg . Yg
Ieqj: Vg . Yg

3. Generadores controlados: Introducen términos siempre que la rama que los controla contenga un generador de tensión independiente. Las siguientes figuras ilustran los distintos casos posibles:

a) Generador de corriente controlado por tensión:

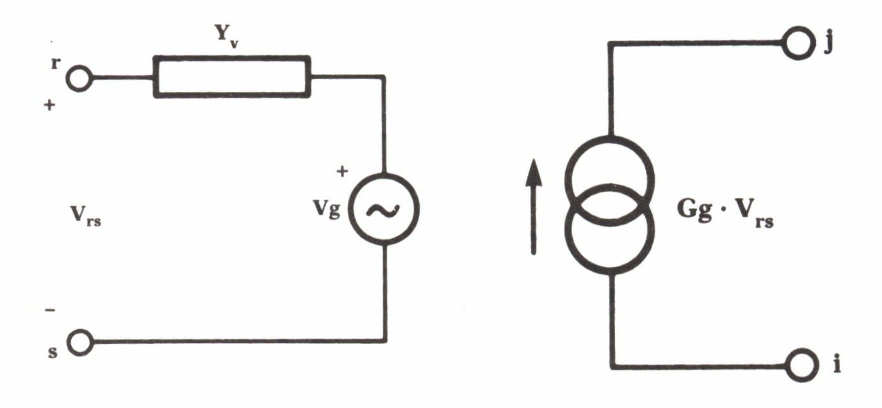


Fig. 21. Generador de corriente controlado por tensión.

Ieqi: –Gg . Vg
Ieqj: Gg . Vg

b) Generador de corriente controlado por corriente:

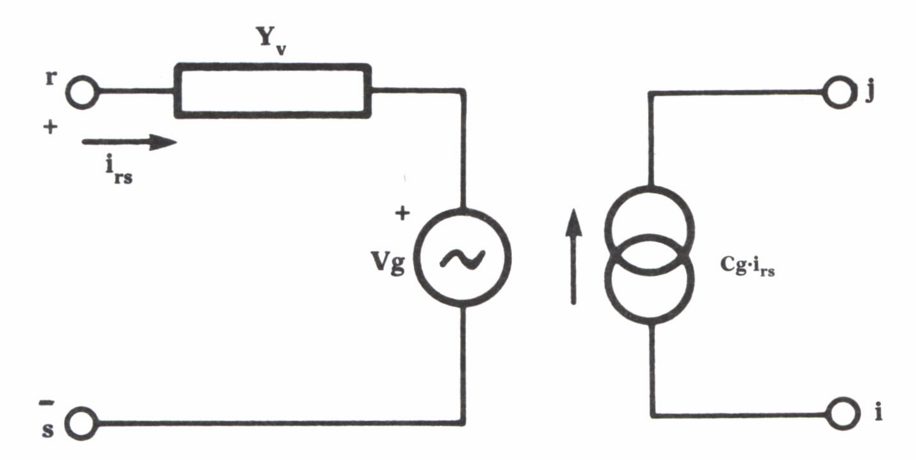


Fig. 22. Generador de corriente controlado por corriente.

Ieqi: –Cg . Vg . Yv

Ieqj: Cg . Vg . Yv

c) Generador de tensión controlado por corriente:

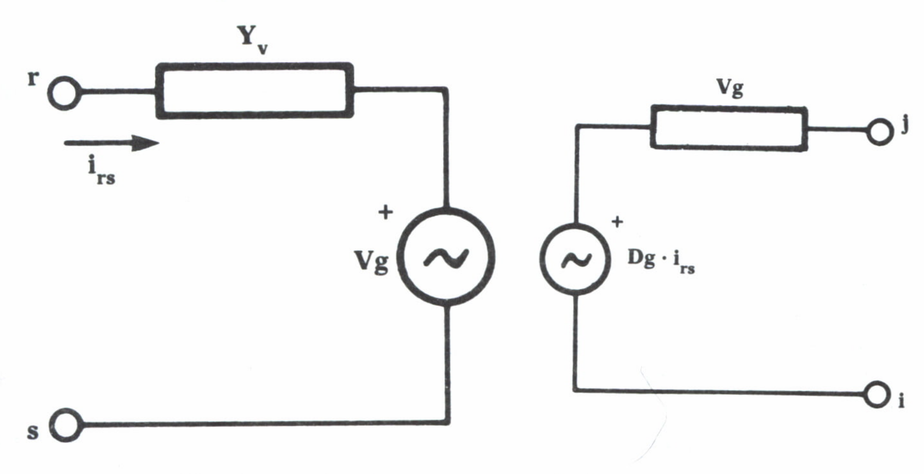


Fig. 23. Generador de tensión controlado por corriente.

Ieqi: –Dg . Yg . Vg . Yv

Ieqj: Dg . Yg . Vg . Yv

d) Generador de tensión controlado por tensión:

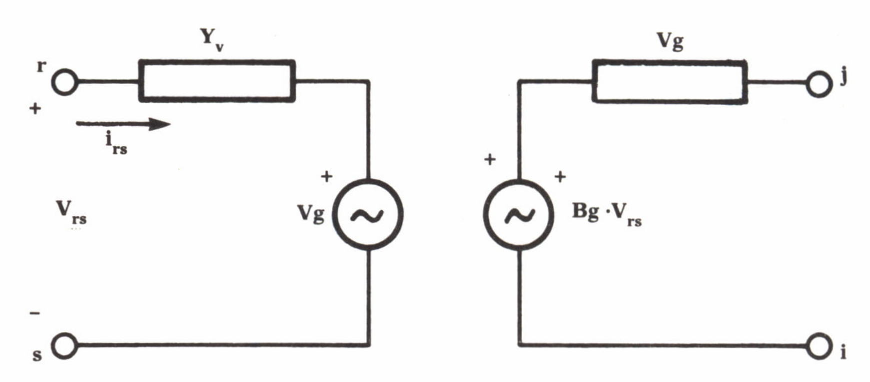


Fig. 24. Generador de tensión controlado por tensión.

Ieqi: –Bg . Yg . Vg
Ieqj: g . Yg . Vg



PROGRAMA EJEMPLO

A continuación se va a presentar un programa que realiza las siguientes funciones:

1. Entrada de datos desde el teclado.
2. Construcción de la matriz de admitancia nodal reducida y el vector de generadores equivalentes.
3. Presentación de ambos.

Los tipos de componentes que acepta son:

- Resistencias
- Bobinas
- Condensadores
- Generadores independientes de corriente.
- Generadores independientes de tensión con admitancia interna resistiva.
- Generadores de corriente controlados por la tensión o la corriente en una rama sin generadores.
- Generadores de tensión con impedancia interna resistiva controlados por la corriente o la tensión en una rama sin generadores.

Con estos componentes se pueden representar más del 90% de los circuitos prácticos, por lo que no suponen una restricción muy grande. Se

pueden realizar fácilmente los cambios necesarios para soportar componentes de otro tipo.

El programa se halla profusamente comentado, con el fin de que el lector pueda analizarlo sobre el mismo listado. No obstante, conviene aquí aclarar algunos puntos que pueden quedar oscuros:

- La entrada de datos se realiza de forma interactiva, pidiéndose el parámetro que en cada momento se debe dar para cada componente de forma explícita.
- Los generadores de corriente se consideran con origen en el nodo del cual parte la corriente, y con destino en el nodo al que dicha corriente se suministra.
- La matriz se presenta en forma compleja, con los términos inductivos separados de los capacitivos, para facilitar la labor a los programas de análisis que se presentarán en el capítulo siguiente, a la vez que se puede imprimir en una forma más compacta.

El listado del programa es el siguiente:

```
Program Entrada_de_Circuito;

(* Este programa realiza la entrada de datos de un
      circuito que contenga dispositivos de los tipos
      siguientes:

      - Resistencias
      - Bobinas
      - Condensadores
      - Generadores independientes de corriente.
      - Generadores independientes de tensión con
            admitancia interna resistiva.
      - Generadores de corriente controlados por la
            tensión o la corriente en una rama sin
            generadores.
      - Generadores de tensión con impedancia interna
            resistiva controlados por la corriente
            o la tensión en una rama sin generadores
   Se necesita también que todos los generadores
      independientes se hallen en fase. Esta fase se
      tomará como orígen. *)

(*===================================================*)
(*= PROCEDIMIENTOS PARA LA  ENTRADA  DEL  CIRCUITO =*)
(*===================================================*)

(* Constantes *)
```

```
Const

  Max_Nodos = 20;
              (* Número máximo de nodos del circuito *)


(* Tipos del sistema *)

Type

(* Tipo de matriz que contendrá las admitancias del
       circuito *)
(* El tercer índice indica el tipo de componente de
   cara al análisis en el dominio de la frecuencia:
    -1: Componente coeficiente de 1/jw   (Bobina)
     0: Componente independiente         (Resistencia)
     1: Componente coeficiente de jw     (Condensador)
     *)

  Circuito    = Array [0..Max_Nodos,0..Max_Nodos,
                       -1..1] of real;

(* Tipo de matriz para los generadores equivalentes *)

  Generadores = Array[0..Max_Nodos] of real;

(* Procedimiento para la entrada del circuito
   Los parámetros que usa son:
   X: Matriz de admitancias del circuito.
         Parámetro de salida del procedimiento.
   Y: Matriz de generadores del circuito.
         Parámetro de salida del procedimiento.
   N: Número de nodos del circuito.
         Parámetro de entrada.
   *)

Procedure Entrada_del_Circuito ( var Y: Circuito;
                                 var I: Generadores);
var
  Componente       : Char;      (* Tipo de componente *)
  N1,                    (* Nodo origen del componente *)
  N2,                   (* Nodo destino del componente *)
  NC1,          (* Nodo origen del elemento controlado *)
  NC2,         (* Nodo destino del elemento controlado *)
  TipoG            : Integer;
                         (* Tipo Generador controlado *)
  ValorComponente,    (* Valor componente en la rama *)
  ValorGenerador,      (* Valor generador controlado *)
```

```
  GGenerador       : Real;
                 (* Valor admitancia interna generador *)
  Error            : Boolean;       (* Mala entrada *)

(* Procedimiento interno auxiliar para entrada de un
   generador de tensión *)

procedure Gen_Tension(var N1,N2: integer;
                      var valor,G_serie : real);

begin
  writeln('Datos del generador:');
  write('Terminal negativo conectado al nudo..>');
  readln(N1);
  write('Terminal positivo conectado al nudo..>');
  readln(N2);
  write('Valor del generador.................>');
  readln(valor);
  write('Valor del componente en serie.......>');
  readln(G_serie);
  G_Serie := 1/G_Serie;
       (* Por ser la entrada impedancia y requerir el
                              análisis admitancias *)
  Valor:=Valor * G_Serie;
                 (* Generador equivalente de corriente *)
end;

(* Procedimiento interno auxiliar para entrada de un
                              generador de corriente *)

procedure Gen_Corriente(var N1,N2: integer;
                        var valor : real);
begin
  writeln('Datos del generador:');
  write('Terminal origen conectado al nudo...>');
  readln(N1);
  write('Terminal destino conectado al nudo...>');
  readln(N2);
  write('Valor del generador.................>');
  readln(valor);
end;


(* Procedimiento interno y auxiliar para la entrada de
   un componente pasivo.  Realiza también la entrada de
   los generadores controlados asociados a la rama que
   se introduce.  *)

procedure Entrada_Componente (C: integer);
```

```
var
  Control           : char;

begin
  TipoG:=0;
  Error:=False;
  writeln('Datos del componente:');
  write('Terminal positivo conectado al nodo..>');
  readln(N1);
  write('Terminal negativo conectado al nodo..>');
  readln(N2);
  write('Valor del componente................>');
  readln(ValorComponente);

(* En caso de que el componente que se introduce sea
   una resistencia o una bobina se debe invertir su
   valor, pues la matriz que calculamos es de
   admitancias *)

  if ((c= 0) or (c = -1)) then ValorComponente:=
                                  1/ValorComponente;

(* Se procede ahora a la entrada de los parámetros del
              generador controlado si es que existe *)

  Control:=' ';
  while not (Control in ['N','S']) do
    (* Repetir la pregunta mientras no se conteste "S"
                                                o "N" *)
     begin
      write('Controla algún generador?...........>');
      readln(Control);
    end;
  if Control ='S' then
    begin
     write('Variable que lo controla (I/V).......>');
     readln(Control);

     case control of
       'V': TipoG:=1;
       'I': TipoG:=2;
       else Error:=True;
     end;

     write('Tipo de generador (I/V).............>');
     readln(Control);

     case Control of
       'V': begin
```

```
                TipoG:=TipoG + 10;
                Gen_Tension(NC1, NC2, ValorGenerador,
                                      GGenerador);
             end;
         'I': begin
                TipoG:=TipoG + 20;
                Gen_Corriente(NC1, NC2,ValorGenerador);
             end;
         else
             Error:=True;
       end;
     end;

(* Si no ha habido error en la entrada de datos se
                              actualizan las matrices *)

   If not Error then
     begin
        (* Actualización de la matriz de admitancias *)
       Y[N1,N1,C]:=Y[N1,N1,C]+ValorComponente;
       Y[N2,N2,C]:=Y[N2,N2,C]+ValorComponente;
       Y[N1,N2,C]:=Y[N1,N2,C]-ValorComponente;
       Y[N2,N1,C]:=Y[N2,N1,C]-ValorComponente;
     end;

(* Actualización de las matrices por la presencia de
                              generadores controlados *)

       if TipoG div 10 = 1 then
         begin
     (* Generador de tensión. Lleva admitancia serie *)
           Y[NC1,NC1,0]:=Y[NC1,NC1,0]+GGenerador;
           Y[NC2,NC2,0]:=Y[NC2,NC2,0]+GGenerador;
           Y[NC1,NC2,0]:=Y[NC1,NC2,0]-GGenerador;
           Y[NC2,NC1,0]:=Y[NC2,NC1,0]-GGenerador;
         end;
       if TipoG mod 10 = 1 then
         begin
                (* Generador controlado por tensión *)
           Y[NC1,N1,0]:=Y[NC1,N1,0]+ValorGenerador;
           Y[NC2,N2,0]:=Y[NC2,N2,0]+ValorGenerador;
           Y[NC1,N2,0]:=Y[NC1,N2,0]-ValorGenerador;
           Y[NC2,N1,0]:=Y[NC2,N1,0]-ValorGenerador;
         end;
       if TipoG mod 10 = 2 then
         begin
         (* Generador controlado por corriente. Hay que
            multiplicar por la admitancia del nudo
            controlador *)
```

```
         Y[NC1,N1,0]:=Y[NC1,N1,0]+ValorGenerador*
                                   ValorComponente;
         Y[NC2,N2,0]:=Y[NC2,N2,0]+ValorGenerador*
                                   ValorComponente;
         Y[NC1,N2,0]:=Y[NC1,N2,0]-ValorGenerador*
                                   ValorComponente;
         Y[NC2,N1,0]:=Y[NC2,N1,0]-ValorGenerador*
                                   ValorComponente;
        end;
    end
    else writeln('Entrada equivocada. ',
                 'Repítala, por favor.');
end;

(* Procedimiento de entrada de componentes *)

begin

(* Se presenta un recordatorio de los tipos de
   componentes que se pueden introducir.  Para acabar
   la entrada de componentes, pulsar la "Q" *)

  writeln('Tipos posibles de componentes:');
  writeln('  - R: Resistencia.');
  writeln('  - L: Bobina.');
  writeln('  - C: Condensador.');
  writeln('  - V: Generador indep. de tensión.');
  writeln('  - I: Generador indep. de corriente.');
  writeln('  - Q: Fin del circuito');

  repeat
    WriteLn;
    Write('Introduzca el tipo de componente.....>');
    ReadLn(Componente);
        (* Espera por el tipo de componente deseado *)

(* Ahora se efectúan las operaciones necesarias para
   cada componente *)

    Case Componente of

(* El tratamiento de R, L y C es análogo, y se realiza
        en el procedimiento   "Entrada_Componente"  *)
      'R' : Entrada_Componente(0);
      'L' : Entrada_Componente(-1);
      'C' : Entrada_Componente(1);

(* Entrada de generador de tensión *)
```

```
      'V' : begin
              Gen_Tension(NC1, NC2, ValorGenerador,
                                        GGenerador);
             (* Introducción de la admitancia serie *)
              Y[NC1,NC1,0]:=Y[NC1,NC1,0]+GGenerador;
              Y[NC2,NC2,0]:=Y[NC2,NC2,0]+GGenerador;
              Y[NC1,NC2,0]:=Y[NC1,NC2,0]-GGenerador;
              Y[NC2,NC1,0]:=Y[NC2,NC1,0]-GGenerador;
              (* Efecto en el vector de generadores *)
              I[NC1]:=I[NC1]-ValorGenerador;
              I[NC2]:=I[NC2]+ValorGenerador;
            end;

(* Entrada de generador de corriente *)
      'I' : begin
              Gen_Corriente(NC1, NC2, ValorGenerador);
              I[NC1]:=I[NC1]-ValorGenerador;
              I[NC2]:=I[NC2]+ValorGenerador;
            end;
      end;
    until Componente='Q'
end;


(*=====================================================*)
(*FIN DE LOS PROCEDIMIENTOS DE ENTRADA DEL CIRCUITO *)
(*=====================================================*)


(* Variables globales *)

Var

(* Matriz de admitancias del circuito *)

  Yn         : Circuito;

(* Matriz de generadores equivalentes *)

  Ieq        : Generadores;

(* Número de nodos del circuito *)

  NNodos     : Integer;

(* Variables auxiliares *)

  i,j,k      : Integer;
```

```
(*=====================================================*)
(*====== RUTINAS AUXILIARES DE VISUALIZACION =======*)
(*=====================================================*)

(* Visualizar la matriz de admitancias *)

Procedure Ver_Matriz_Admitancias(var Y: Circuito);
var i,j: integer;

begin
  Writeln;
  Writeln('Matriz de admitancias del circuito');
  Writeln('----------------------------------');
  Writeln;
  for i:=1 to NNodos do
    for j:=1 to NNodos do begin
      WriteLn('Y(',i,',',j,') = ',Y[i,j,-1],' 1/jw');
      WriteLn(' + ',Y[i,j,0],' + ',Y[i,j,1],' jw');
      end;
end;

(* Visualizar matriz de generadores independientes *)

Procedure Ver_Generadores( var I: Generadores);

var j:integer;

begin
  Writeln;
  Writeln('Matriz de generadores independientes');
  Writeln('------------------------------------');
  Writeln;
  for j:=1 to NNodos do
    WriteLn('I(',j,') = ',I[j]);
end;


(*=====================================================*)
(*= FIN DE LAS RUTINAS AUXILIARES DE VISUALIZACION =*)
(*=====================================================*)



(* Programa Principal *)

BEGIN

(* Borrado de la matriz de admitancias *)
```

```
  for i:=0 to Max_Nodos do
    for j:=0 to Max_Nodos do
      for k:=-1 to 1 do
        Yn[i,j,k]:=0;
(* Borrado matriz de generadores independientes *)
  for i:=0 to Max_Nodos do
    Ieq[i]:=0;
(* Petición del número de nodos, comprobando si
   rebasa el máximo permitido *)
  repeat
    WriteLn('Introduzca el número de nodos del');
    Write(' circuito (Sin contar el de masa)...>');
    ReadLn(NNodos);
    If NNodos > Max_Nodos
      then writeln ('No es posible procesar tantos',
                    ' nodos con este programa');
    until NNodos <= Max_Nodos;
  Writeln;
  Writeln('ATENCION: Ponga su teclado en mayúsculas');
  Writeln;
(* Se llama a la rutina de entrada del circuito *)
  Entrada_del_Circuito(Yn,Ieq);
(* Se visualiza la matriz de admitancias calculada y
   el vector de generadores independientes *)
  Ver_Matriz_Admitancias(Yn);
  Ver_Generadores(Ieq);
END.
```

Para ilustrar el funcionamiento del programa se va a aplicar al cálculo de la matriz de admitancias reducida del siguiente circuito:

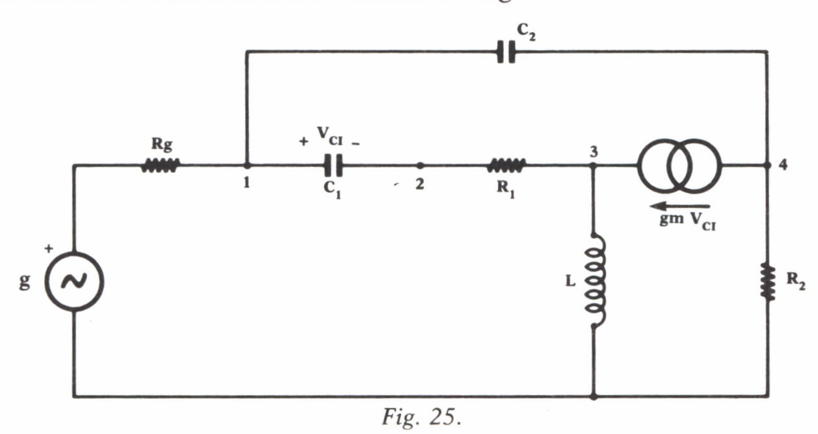


Fig. 25.

En el circuito se han marcado los nodos con el número que se les va a dar al ejecutar el programa. Los valores asignados a los componentes son:

Vg = 1 V	C1 = 10 nF	R1 = 1 KΩ
Rg = 50 Ω	C2 = 10 pF	R2 = 100 Ω
gm = 0.3	L = 2 mH	

A continuación se presenta el proceso tal y como se vería en la pantalla del ordenador:

```
Introduzca el número de nodos del
 circuito (Sin contar el de masa)...>4

ATENCION: Ponga su teclado en mayúsculas

Tipos posibles de componentes:
  - R: Resistencia.
  - L: Bobina.
  - C: Condensador.
  - V: Generador indep. de tensión.
  - I: Generador indep. de corriente.
  - Q: Fin del circuito

Introduzca el tipo de componente.....>V
Datos del generador:
Terminal negativo conectado al nudo..>0
Terminal positivo conectado al nudo..>1
Valor del generador..................>1
Valor del componente en serie........>50

Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>1
Terminal negativo conectado al nodo..>2
Valor del componente.................>10E-9
Controla algún generador?............>S
Variable que lo controla (I/V).......>V
Tipo de generador (I/V)..............>I
Datos del generador:
Terminal origen conectado al nudo....>4
Terminal destino conectado al nudo...>3
Valor del generador..................>0.3

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>3
Valor del componente.................>1E3
Controla algún generador?............>N

Introduzca el tipo de componente.....>L
```

```
Datos del componente:
Terminal positivo conectado al nodo..>3
Terminal negativo conectado al nodo..>0
Valor del componente.................>2E-3
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>4
Terminal negativo conectado al nodo..>0
Valor del componente.................>100
Controla algún generador?............>N
Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>4
Terminal negativo conectado al nodo..>1
Valor del componente.................>10E-2
Controla algún generador?............>N

Introduzca el tipo de componente.....>Q


Matriz de admitancias del circuito
----------------------------------

Y(1,1) =   0.0000000000E+00 1/jw
 +   2.0000000000E-02 +   1.0000001000E-01 jw
Y(1,2) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-08 jw
Y(1,3) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(1,4) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-01 jw
Y(2,1) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-08 jw
Y(2,2) =   0.0000000000E+00 1/jw
 +   1.0000000000E-03 +   1.0000000000E-08 jw
Y(2,3) =   0.0000000000E+00 1/jw
 +  -1.0000000000E-03 +   0.0000000000E+00 jw
Y(2,4) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(3,1) =   0.0000000000E+00 1/jw
 +  -3.0000000000E-01 +   0.0000000000E+00 jw
Y(3,2) =   0.0000000000E+00 1/jw
 +   2.9900000000E-01 +   0.0000000000E+00 jw
Y(3,3) =   5.0000000000E+02 1/jw
 +   1.0000000000E-03 +   0.0
```


```
Y(3,2) =   0.0000000000E+00 1/jw
 +   2.9900000000E-01 +   0.0000000000E+00 jw
```


```
Y(3,3) =   5.0000000000E+02 1/jw
 +   1.0000000000E-03 +   0.0000000000E+00 jw
```

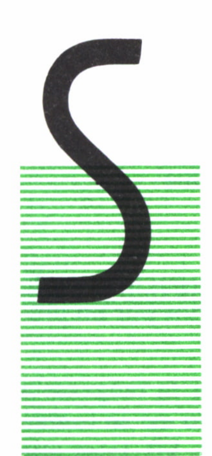
```
Y(3,4) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(4,1) =    0.0000000000E+00 1/jw
 +    3.0000000000E-01 +   -1.0000000000E-01 jw
Y(4,2) =    0.0000000000E+00 1/jw
 +   -3.0000000000E-01 +    0.0000000000E+00 jw
Y(4,3) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(4,4) =    0.0000000000E+00 1/jw
 +    1.0000000000E-02 +    1.0000000000E-01 jw

Matriz de generadores independientes
------------------------------------

I(1) =    2.0000000000E-02
I(2) =    0.0000000000E+00
I(3) =    0.0000000000E+00
I(4) =    0.0000000000E+00
```

Como ya se especificaba en los comentarios del listado, cada elemento de la matriz de admitancias tiene tres componentes, que se deben al efecto de las bobinas, resistencias y condensadores, respectivamente, que hay en el circuito. La matriz de generadores independientes es real. Esto implica que, en caso de haber varios en el circuito, todos tendrán la misma fase, que se tomará como origen de fases para el resto de corrientes y tensiones.

ANALISIS DE CIRCUITOS EN CONTINUA 3



EGUN lo visto en el capítulo anterior, para analizar un circuito es indispensable realizar el paso de los componentes físicos y su interconexionado a un sistema de ecuaciones. Esto se logra mediante

• las relaciones corriente-tensión de los componentes lineales (o el modelo linealizado de los circuitos no lineales), y

• los lemas de Kirchhoff, que fijan las características de conexionado entre los primeros,

resumidas mediante unas reglas para el análisis nodal, con el fin de implementarlas con facilidad por ordenador.

Con esto se obtiene, en el caso más general, un sistema de ecuaciones diferenciales que representan el funcionamiento *completo* de la red.

El programa de análisis descompone normalmente el problema en varios casos:

• Análisis en continua.
• Análisis en alterna.
• Análisis transitorio.

El presente capítulo va a abordar el primero de estos problemas, mientras que los dos restantes se tratarán en los siguientes.

Se va a dividir en dos partes fundamentales:

• Desarrollo de las matemáticas necesarias para llevar a cabo el análisis en corriente continua.

• Programa sencillo para que el lector pueda comprobar una implementación muy sencilla de los métodos matemáticos ilustrados en el capítulo.

EL PROBLEMA DE LOS COMPONENTES NO LINEALES

El análisis de un circuito en continua es muy sencillo en el caso de circuitos lineales. En este caso, las bobinas son un cortocircuito y los condensadores un circuito abierto, con lo cual la red se simplifica enormemente. Solamente es necesario considerar resistencias y generadores.

El mayor interés de este tipo de análisis está en el caso de circuitos no lineales. Es así porque, normalmente, los datos de funcionamiento que se obtengan para los componentes no lineales (punto de trabajo) van a fijar los parámetros para el circuito equivalente en alterna, donde normalmente nos interesan más los resultados.

Un dispositivo no lineal es aquel cuya relación corriente-tensión sigue una variación no lineal, que se puede representar de la forma:

$$v = f(i)$$

Estos casos darían lugar a un sistema de ecuaciones diferenciales no lineales difícil de resolver. Por esto lo que se hace es resolverlos por un procedimiento iterativo de la siguiente forma:

1. Se linealiza el circuito, aproximando la función por la tangente a ella en un determinado punto elegido arbitrariamente. Esto es: se toman como tensión y corriente unos valores que llamaremos I0, V0, tales que satisfagan que

$$V0 = f(I0)$$

Con ellos se aproxima la función por:

$$v = V0 + f'(I0)\ (i - I0) = Veq + Req\ i$$

$Veq = V0 - f'(I0).I0$
$Req = f'(I0)$

que es la ecuación de un circuito como el de la siguiente figura:

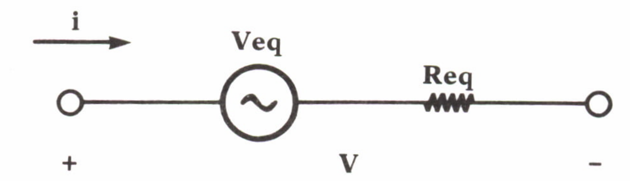


Fig. 26. Circuito linealizado de un componente no lineal definido en tensión.

En caso de que el circuito no lineal nos viniese dado por una fórmula cuya variable independiente fuera la tensión y no resultase ser fácilmente despejable, el procedimiento sería análogo:

$i = f(v)$
$I0 = f(V0)$

$$i = I0 + f'(V0)\,(v - V0) = Ieq + \frac{v}{Req} \quad (1)$$

donde

$$Ieq = I0 - f'(V0).V0$$

y

$$Req = \frac{1}{f'(V0)}$$

cuyo circuito equivalente sería

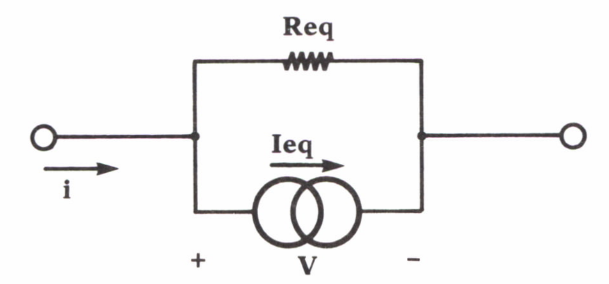


Fig. 27. Circuito linealizado de un componente no lineal definido en corriente.

2. Una vez linealizado se resuelve este circuito en continua. Con ello se obtienen unos valores de tensión y corriente para nuestro dispositivo, que no tienen por qué coincidir con los introducidos «a priori».

3. Con los nuevos valores obtenidos se calcula el nuevo circuito equivalente lineal del dispositivo. Este ciclo se repite hasta que la diferencia entre una solución y la siguiente sea menor que el margen de error que se pueda tolerar.

A continuación se van a presentar las linealizaciones de algunos de los dispositivos no lineales más corrientes.

EQUIVALENTES LINEALES DE ALGUNOS DISPOSITIVOS

Diodo de unión PN

Sin entrar en efectos de segundo orden, el modelo más simple del diodo que se puede establecer es el siguiente:

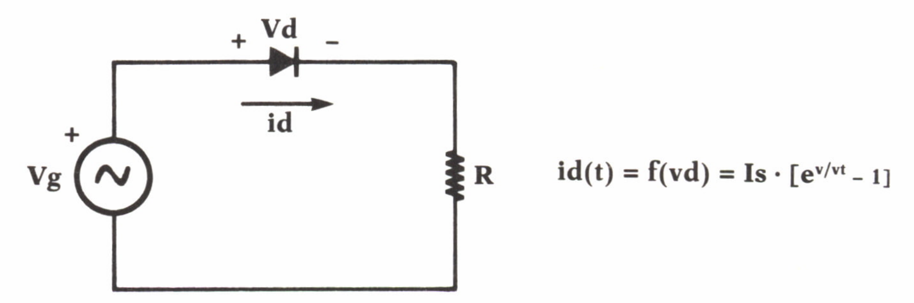


Fig. 28. Diodo de unión PN y su relación corriente tensión.

Las constantes de esta ecuación tienen el siguiente significado:

- Is: Corriente de saturación. La podemos hallar en las características del catálogo, por representar la corriente que circula por el diodo bajo polarización inversa.
- Vt = KT/q , en donde:

K: Constante de Boltzmann: $1.38^{-23} \times 10$ Jul/Kelvin
T: Temperatura en Kelvin
q: Carga del electrón: 1.6×10^{-19} Q

Con todo esto, a temperatura ambiente,

$$Vt = 0.025 \text{ V.}$$

Junto con la ecuación del diodo tenemos la ecuación obtenida por la ley de Kirchhoff de tensiones:

$$Vg = vd + id \cdot R \quad (3)$$

El procedimiento más corriente consiste en realizar una construcción gráfica (ver figura 29). Sobre la curva que representa el comportamiento del diodo se traza la recta que fijan las condiciones externas al diodo (cuya ecuación nos ha dado la ley de Kirchhoff), llamada *recta de carga*. El punto de intersección de ambas es la solución del problema.

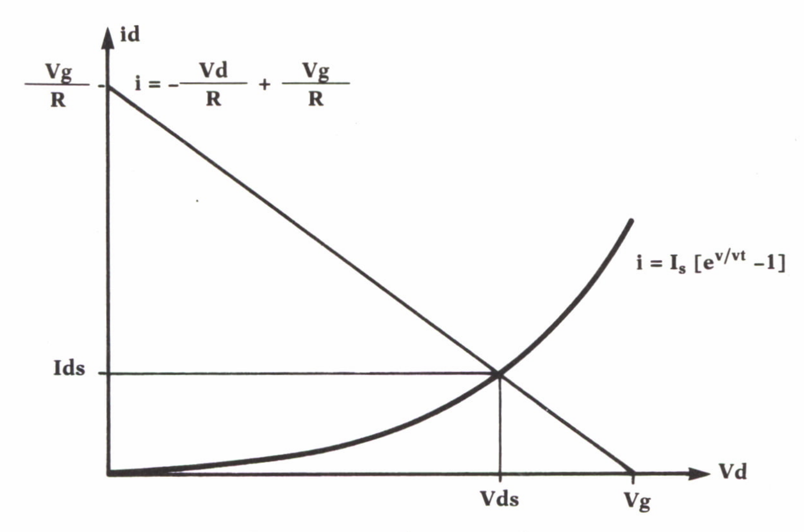


Fig. 29. Resolución gráfica.

Para emplear el procedimiento iterativo expuesto en este capítulo se debe linealizar la ecuación del componente. Su derivada es:

$$f'(v) = \frac{Is}{Vt} (e^{V/Vt} - 1)$$

Tomemos ahora una solución arbitraria que denominaremos Id0, Vd0, tales que verificarán que:

$$Id0 = Is\ (e^{Vd0/Vt} - 1)$$

Particularizando la derivada en este punto se obtiene:

$$f'(V0) = \frac{Is}{Vt} (e^{V0/Vt} - 1) = gm$$

La ecuación linealizada del diodo será, según (1):

$$id = Ido + gm\ (\ vd\ Vd0\ )\ (4)$$

Sustituyendo el valor de id que se obtiene de esta ecuación en la (3), se resuelve el problema en esta primera aproximación:

$$Vg = (Id0 + gm\ (vd\ Vd0))\ R + vd$$
$$= (Id0\ gm\ Vd0)\ R + (gm + 1)\ vd$$

$$vd = \frac{Vg - (Id0 - gm\ Vd0)\ R}{gm\ R + 1} = Vd1$$

Y sustituyendo este valor de vd en (2):

$$Id1 = Is\ (e^{Vd1/Vt} - 1)$$

se obtiene una nueva solución aproximada Id1, Vd1, con la cual se repetiría el proceso.



Transistor bipolar

El modelo más sencillo para el transistor es el de Ebers-Moll, que nos proporciona un circuito equivalente como el de la figura:

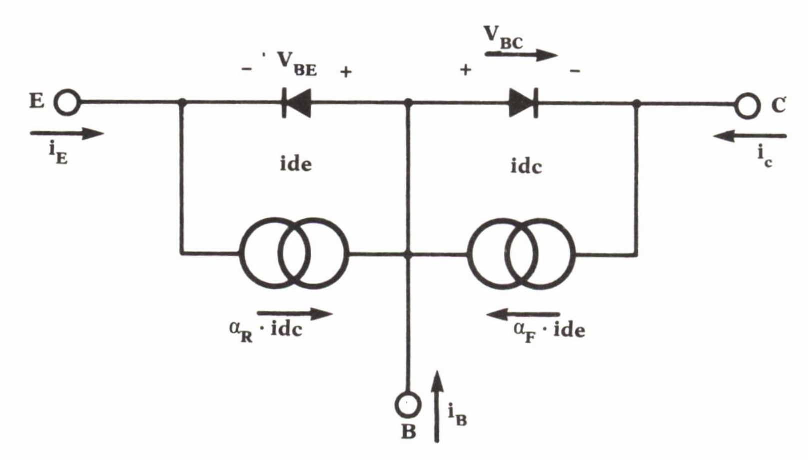


Fig. 30. Circuito equivalente del transistor bipolar NPN.

Este es el circuito general, empleado para el análisis transitorio, y en ese capítulo se volverá sobre él. Aquí, de momento, los condensadores se eliminarán por ser un circuito abierto en continua.

Aplicando la ley de Kirchhoff de corrientes en los tres nudos Emisor (E), Colector (C) y Base (B), junto con la ecuación del diodo, citada anteriormente, se obtiene:

$$ie = ide + \alpha r\ .\ idc =$$

$$= Ies\ (e^{Vbe/Vt} - 1) + \alpha r\ Ics\ (e^{Vbc/Vt} - 1)$$

$$ic = -\ idc + \alpha f\ .\ ide =$$

$$= -\ Ics\ (e^{Vbc/Vt} - 1) + \alpha f\ Ies\ (e^{Vbe/Vt} - 1)$$

$$ib = ic - ie$$

Valores típicos para estos parámetros son:

Ies = Ics = 1 nA
αf = .98
αr = .5

Entre αf y la ganancia en corriente del transistor, β, existe la relación:

$$\alpha f = \frac{\beta}{\beta + 1}$$

En este caso, al haber dos ramas con elementos no lineales, la resolución se complica bastante.

Tomaremos para ilustrar el método a seguir un sencillo circuito con un transistor en configuración de emisor común:

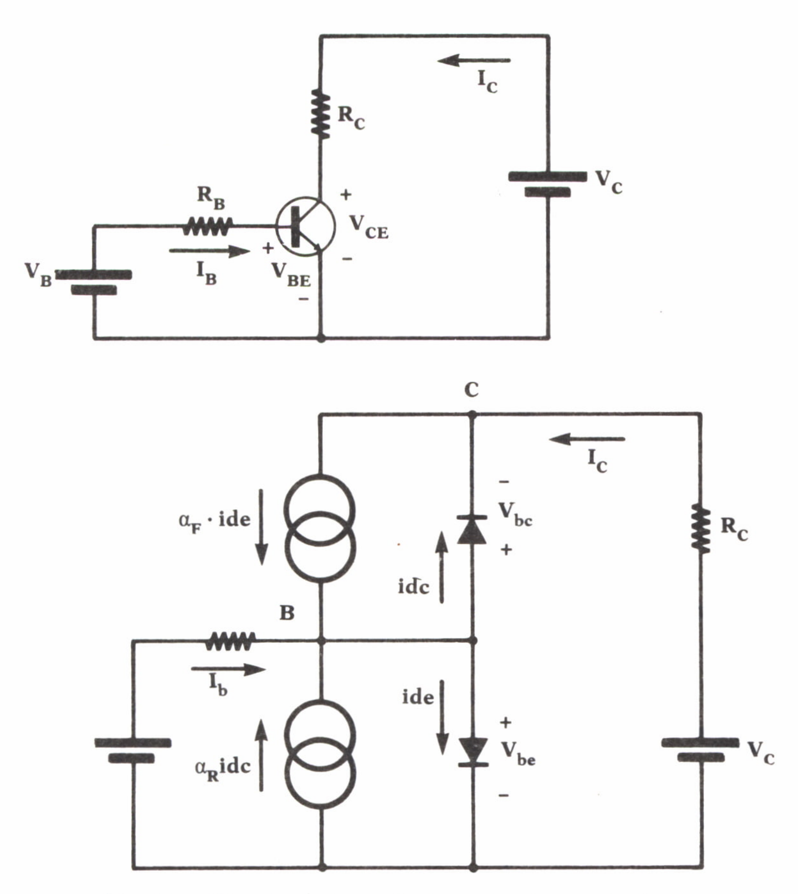


Fig. 31. Circuito bajo análisis y circuito equivalente.

Aplicando la ley de Kirchhoff de corrientes a los nudos de base y colector, se obtiene:

$$Ib + \alpha f \ ide + \alpha r \ idc - ide - idc = 0$$
$$Ic + idc - \alpha f \ ide = 0$$

Operando con esas ecuaciones resulta:

$$(Vgb - Vb)\ (1/Rb) + (\ \alpha f - 1\ )\ ide + (\ \alpha r - 1\ )\ idc = 0 \qquad (5)$$
$$(Vgc - Vc)\ (1/Rc)\ \alpha f \ ide + idc = 0 \qquad (6)$$

Se toma una solución arbitraria para las tensiones base-colector y base-emisor Vbe0 y Vbc0. Aproximando por la tangente a la curva característica de los diodos base-emisor y base-colector, como se hizo en (4), obtenemos:

$$ide = Ide0 + gme\ (vbe - Vbe0) \qquad (7)$$
$$gme = (Ies/Vt)\ (e^{vbe/Vt} - 1)$$
$$idc = Idc0 + gmc\ (vbc - Vbc0) \qquad (8)$$
$$gmc = (Ics/Vt)\ (e^{vbc/Vt} - 1)$$

El circuito equivalente lineal para esta aproximación será:

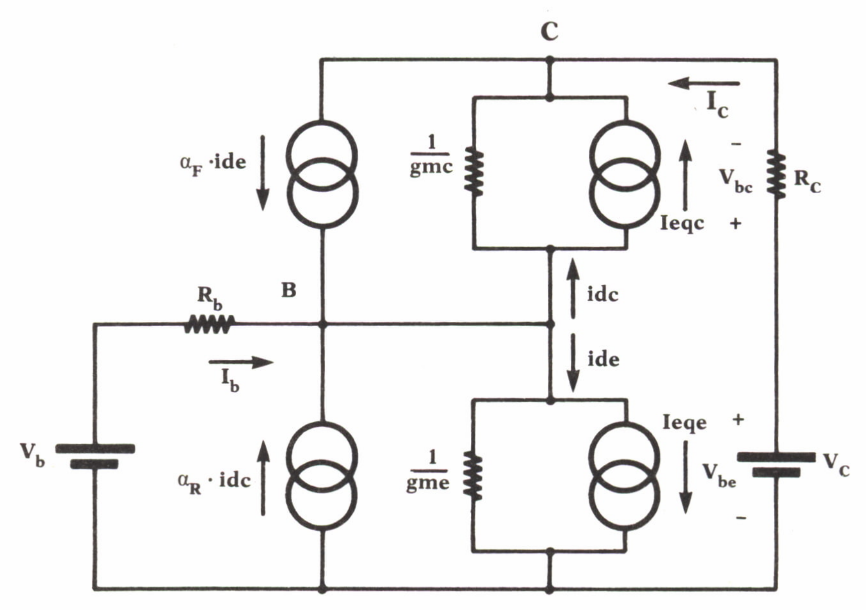


Fig. 32. Circuito linealizado en la primera aproximación.

En el cual se ha hecho:

$$Ieqe = Ide0 - gme\ Vbe0$$
$$Ieqc = Idc0 - gmc\ Vbc0$$

Este circuito lineal se puede resolver por el método del análisis nodal, y obtener unos valores de Vb y Vc. Con ellos podremos calcular la segunda aproximación para las tensiones vbe y vbc, ya que, por construcción del circuito:

$$vbe = Vb$$
$$vbc = Vb - Vc$$

Con esta segunda aproximación se repetiría de nuevo el cálculo completo.



Transistor de efecto de campo

Su modelo más corriente se puede representar:

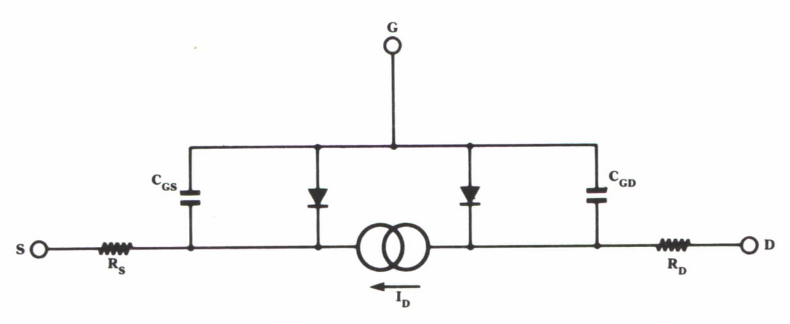


Fig. 33. Modelo en continua del transistor FET.

El valor de la corriente que circula entre drenador y fuente, que llamaremos Id es el siguiente:

$$Id\ Idss \frac{2\ Vds}{Vp} \left(\frac{Vgs}{Vp} - \frac{Vds}{2\ Vp} - 1 \right)$$ Para la zona lineal.

$$Id = Idss \left(1 - \frac{Vgs}{Vp}\right)^2$$ Para saturación.

En esta expresión:

- Vp es la tensión de estrangulamiento o de «Pinch-off». Es el valor de la tensión puerta-fuente por encima del cual la corriente drenador-fuente no aumenta.
- Idss es la corriente de drenador para Vgs = 0.

Para el análisis en continua las capacidades serán un circuito abierto, con lo que desaparecerán del modelo, simplificando así el cálculo.



Transistor MOSFET

Se expone aquí el modelo de primer orden, o de Shichman Hodges:

Fig. 34. Modelo en continua del transistor MOSFET.

La fórmula para ID es la misma que en el transistor FET de unión.

Del mismo modo que en el transistor FET, las capacidades en continua desaparecerán del modelo, simplificando así el cálculo. Vemos como, al desaparecer las capacidades, nos queda la puerta (G) sin conectar a nin-

gún nodo. Por esto es por lo que a los transistores **MOSFET** se les denomina transistores de efecto de campo de puerta aislada.



RESUMEN

El método aquí expuesto puede no llevarnos a la solución en todos los casos que se nos ocurra plantear. Como todo método iterativo, su convergencia hacia la solución del problema sólo está garantizada bajo determinadas condiciones, en las que aquí no entraremos. Baste con decir que, si la solución inicial es lo bastante próxima a la real, el método suele dar resultados muy buenos y en pocas iteraciones. Pero, en caso contrario, puede incluso diverger, alejándonos cada vez más de nuestro objetivo.

En estos casos se emplean otros métodos, como son el método de Newton modificado y el método de Broyden, para cuya consulta se remite al lector interesado a la bibliografía básica que podrá encontrar al final del libro.

Otro problema que se puede plantear es el de Overflow, esto es, encontrar un valor de tensión o de corriente que desborde la capacidad de nuestro ordenador. Esto puede ocurrir, por ejemplo, si se llega a una iteración que presenta un valor demasiado alto para la tensión en un diodo o en la base de un transistor. Este problema se puede resolver fácilmente si se toman unos límites superiores para estos valores, fijados a partir de las condiciones que imponga el circuito en el peor caso.



PROGRAMA EJEMPLO

Para ilustrar los métodos expuestos en este capítulo se presenta un programa capaz de analizar circuitos lineales en continua y circuitos con diodos.

Este programa incorpora para el análisis los siguientes procedimientos:

• Entrada de un circuito y visualización de las matrices que lo definen. Este procedimiento es el mismo que se presentó en el capítulo anterior, en el que se ha suprimido la parte de entrada de inductancias y condensadores, no necesaria en este caso, con lo cual la matriz de admitancias interna se reduce en una dimensión.

• Resolución de un sistema de ecuaciones por el método de descomposición L-U. Estos procedimientos se detallan en el capítulo «Resolución de sistemas de ecuaciones», donde forman parte integrante de los programas que allí se estudian.

• Procedimientos para la entrada de los diodos, linealización y análisis del circuito. Son los propios del capítulo presente, y, por ello, se presentan con un gran número de comentarios para su fácil seguimiento. Por el contrario, en los programas ya tratados éstos se han suprimido, para concentrar la atención en la parte original que aquí se pretende analizar.

Conviene aclarar, al margen de los comentarios del programa, los siguientes aspectos:

• Para la tensión en los diodos se han tomado unos límites máximo y mínimo. Debido a la característica exponencial del diodo, para tensiones grandes, el valor que toma la función exponencial involucrada en el cálculo es enorme, pudiendo producir «Overflow» en nuestro ordenador. Del mismo modo puede existir «Underflow» para tensiones negativas, pues el valor de la exponencial es muy pequeño.

Por estas razones se han tomado unos límites superior e inferior con los cuales se truncan dichos cálculos. Estos valores dependerán de la capacidad de manejo de datos del compilador que estemos usando.

• Las rutinas de entrada se han mantenido con una estructura básica igual a la expuesta en el capítulo «Fundamentos del diseño de circuitos asistido por ordenador», en la página 13. En este caso se usa sólo la parte correspondiente a los términos reales, desapareciendo, por tanto, la tercera dimensión de la matriz que indicaba el tipo de componente introducido. El usuario debe tener esto en cuenta, sustituyendo en el circuito que desee analizar las bobinas por cortocircuitos y los condensadores por circuitos abiertos, realizando así la asignación de nudos.

• Los valores que se asignen a los diodos como condiciones iniciales son sumamente críticos. No se deben asignar tensiones nulas, por producir errores en la resolución del sistema de ecuaciones. Valores en torno a 0.4 voltios pueden considerarse recomendables, si bien pueden necesitarse varios intentos para llegar a obtener un resultado aceptable.

Una vez establecidas estas consideraciones, he aquí el programa:

```
Program Solucion_en_Continua;
(* Este programa hace un análisis en continua de
   circuitos:
      - Lineales
      - Con diodos.
   Las restricciones impuestas en el programa de
   entrada de datos son válidas también en este caso*)
(*================================================*)
(*====== ENTRADA DE UN CIRCUITO EN CONTINUA =======*)
(*================================================*)
```

```
(* Constantes *)

Const

  Max_Nodos = 20;
            (* Número máximo de nodos del circuito *)


(* Tipos del sistema *)

Type

(* Tipo de matriz que contendrá las admitancias del
   circuito. En este caso es del mismo tipo que la
   matriz del sistema de ecuaciones *)

  Sistema     = Array [0..Max_Nodos,0..Max_Nodos]
                      of real;

(* Tipo de matriz para los generadores equivalentes *)

  Vector      = Array[0..Max_Nodos] of real;

(* Procedimiento para la entrada del circuito
   Los parámetros que usa son:
   X: Matriz de admitancias del circuito.
      Parámetro de salida del procedimiento.
   Y: Matriz de generadores del circuito.
      Parámetro de salida del procedimiento.
   N: Número de nodos del circuito.
      Parámetro de entrada.
   *)

(* Procedimiento para la entrada del circuito
   Los parámetros que usa son:
   X: Matriz de admitancias del circuito.
         Parámetro de salida del procedimiento.
   Y: Matriz de generadores del circuito.
         Parámetro de salida del procedimiento.
   N: Número de nodos del circuito.
         Parámetro de entrada.
   *)

Procedure Entrada_del_Circuito ( var Y: Sistema;
                                 var I: Vector);
var
  Componente      : Char;     (* Tipo de componente *)
```

```
  N1,                    (* Nodo origen del componente *)
  N2,                   (* Nodo destino del componente *)
  NC1,         (* Nodo origen del elemento controlado *)
  NC2,        (* Nodo destino del elemento controlado *)
  TipoG              : Integer;
                          (* Tipo Generador controlado *)
  ValorComponente,    (* Valor componente en la rama *)
  ValorGenerador,      (* Valor generador controlado *)
  GGenerador          : Real;
              (* Valor admitancia interna generador *)
  Error               : Boolean;         (* Mala entrada *)

(* Procedimiento interno auxiliar para entrada de un
   generador de tensión *)

procedure Gen_Tension(var N1,N2: integer;
                      var valor,G_serie : real);

begin
  writeln('Datos del generador:');
  write('Terminal negativo conectado al nudo..>');
  readln(N1);
  write('Terminal positivo conectado al nudo..>');
  readln(N2);
  write('Valor del generador..................>');
  readln(valor);
  write('Valor del componente en serie........>');
  readln(G_serie);
  G_Serie := 1/G_Serie;
       (* Por ser la entrada impedancia y requerir el
                               análisis admitancias *)
  Valor:=Valor * G_Serie;
              (* Generador equivalente de corriente *)
end;

(* Procedimiento interno auxiliar para entrada de un
                              generador de corriente *)

procedure Gen_Corriente(var N1,N2: integer;
                        var valor : real);
begin
  writeln('Datos del generador:');
  write('Terminal origen conectado al nudo....>');
  readln(N1);
  write('Terminal destino conectado al nudo...>');
  readln(N2);
```

```
    write('Valor del generador.................>');
    readln(valor);
  end;

  (* Procedimiento interno y auxiliar para la entrada de
     una resistencia.  Realiza también la entrada de los
     generadores controlados asociados a la rama que se
     introduce.  *)

  procedure Entrada_Componente;

  var
    Control          : char;

  begin
    TipoG:=0;
    Error:=False;
    writeln('Datos del componente:');
    write('Terminal positivo conectado al nodo..>');
    readln(N1);
    write('Terminal negativo conectado al nodo..>');
    readln(N2);
    write('Valor del componente................>');
    readln(ValorComponente);

  (* Se invierte el valor de la resistencia por estar
     calculando una matriz de admitancias *)

    ValorComponente:=1/ValorComponente;

  (* Se procede ahora a la entrada de los parámetros del
     generador controlado si es que existe *)

    Control:=' ';
    while not (Control in ['N','S']) do
      (* Repetir la pregunta mientras no se conteste "S"
                                                    o "N" *)
      begin
        write('Controla algún generador?...........>');
        readln(Control);
      end;
    if Control ='S' then
      begin
        write('Variable que lo controla (I/V)......>');
        readln(Control);

        case control of
```

```
      'V': TipoG:=1;
      'I': TipoG:=2;
      else Error:=True;
    end;

    write('Tipo de generador (I/V)..............>');
    readln(Control);
    case Control of
      'V': begin
             TipoG:=TipoG + 10;
             Gen_Tension(NC1, NC2, ValorGenerador,
                                   GGenerador);
           end;
      'I': begin
             TipoG:=TipoG + 20;
             Gen_Corriente(NC1, NC2,ValorGenerador);
           end;
      else
           Error:=True;
    end;
  end;

(* Si no ha habido error en la entrada de datos se
                           actualizan las matrices *)

 If not Error then
   begin
     (* Actualización de la matriz de admitancias *)
    Y[N1,N1]:=Y[N1,N1]+ValorComponente;
    Y[N2,N2]:=Y[N2,N2]+ValorComponente;
    Y[N1,N2]:=Y[N1,N2]-ValorComponente;
    Y[N2,N1]:=Y[N2,N1]-ValorComponente;

(* Actualización de las matrices por la presencia de
                           generadores controlados *)

    if TipoG div 10 = 1 then
      begin
   (* Generador de tensión. Lleva admitancia serie *)
        Y[NC1,NC1]:=Y[NC1,NC1]+GGenerador;
        Y[NC2,NC2]:=Y[NC2,NC2]+GGenerador;
        Y[NC1,NC2]:=Y[NC1,NC2]-GGenerador;
        Y[NC2,NC1]:=Y[NC2,NC1]-GGenerador;
      end;
    if TipoG mod 10 = 1 then
      begin
             (* Generador controlado por tensión *)
        Y[NC1,N1]:=Y[NC1,N1]+ValorGenerador;
```

```
          Y[NC2,N2]:=Y[NC2,N2]+ValorGenerador;
          Y[NC1,N2]:=Y[NC1,N2]-ValorGenerador;
          Y[NC2,N1]:=Y[NC2,N1]-ValorGenerador;
        end;
      if TipoG mod 10 = 2 then
        begin
        (* Generador controlado por corriente. Hay que
           multiplicar por la admitancia del nudo
           controlador *)
          Y[NC1,N1]:=Y[NC1,N1]+ValorGenerador*
                                ValorComponente;
          Y[NC2,N2]:=Y[NC2,N2]+ValorGenerador*
                                ValorComponente;
          Y[NC1,N2]:=Y[NC1,N2]-ValorGenerador*
                                ValorComponente;
          Y[NC2,N1]:=Y[NC2,N1]-ValorGenerador*
                                ValorComponente;
        end;
    end
    else writeln('Entrada equivocada. ',
                 'Repítala, por favor.');
end;

(* Comienzo del procedimiento entrada de componentes*)

begin

(* Se presenta un recordatorio de los tipos de
   componentes que se pueden introducir.  Para acabar
   la entrada de componentes, pulsar la "Q" *)

  writeln('Tipos posibles de componentes:');
  writeln(' ´- R: Resistencia.');
  writeln('  - V: Generador indep. de tensión.');
  writeln('  - I: Generador indep. de corriente.');
  writeln('  - Q: Fin del circuito');

  repeat
    WriteLn;
    Write('Introduzca el tipo de componente.....>');
    ReadLn(Componente);
        (* Espera por el tipo de componente deseado *)

(* Ahora se efectúan las operaciones necesarias para
   cada componente *)

    Case Componente of
```

```
(* El tratamiento de R, L y C es análogo, y se realiza
        en el procedimiento   "Entrada_Componente"  *)
      'R' : Entrada_Componente;

(* Entrada de generador de tensión *)
      'V' : begin
              Gen_Tension(NC1, NC2, ValorGenerador,
                                    GGenerador);
             (* Introducción de la admitancia serie *)
              Y[NC1,NC1]:=Y[NC1,NC1]+GGenerador;
              Y[NC2,NC2]:=Y[NC2,NC2]+GGenerador;
              Y[NC1,NC2]:=Y[NC1,NC2]-GGenerador;
              Y[NC2,NC1]:=Y[NC2,NC1]-GGenerador;
              (* Efecto en el vector de generadores *)
              I[NC1]:=I[NC1]-ValorGenerador;
              I[NC2]:=I[NC2]+ValorGenerador;
            end;

(* Entrada de generador de corriente *)
      'I' : begin

             Gen_Corriente(NC1, NC2, ValorGenerador);
             I[NC1]:=I[NC1]-ValorGenerador;
             I[NC2]:=I[NC2]+ValorGenerador;
            end;
      end;
   until Componente='Q'
end;


(*=====================================================*)
(*FIN DE LOS PROCEDIMIENTOS DE ENTRADA DEL CIRCUITO *)
(*=====================================================*)



(*=====================================================*)
(*=====  RESOLUCION DE UN SISTEMA DE ECUACIONES =====*)
(*=====================================================*)


Procedure Resuelve_Sistema(var A: Sistema;
          var B: vector; Orden: Integer;var X:Vector);

var
  i,j,k  : integer;
  Y      : Vector;
  (* Solución intermedia de LY = B *)
```

```
Begin

(* Marcha directa. Descomposición A = LU *)

  For j:=1 to Orden do
    Begin
      for i:=j to Orden do
        begin              (* Coeficientes matriz L *)
          for k:=1 to j-1 do A[i,j]:=A[i,j]
                                    -A[i,k]*A[k,j];
        end;
      for i:=j+1 to Orden do
        begin
          (* Coeficientes matriz U *)
          for k:=1 to j-1 do A[j,i]:=A[j,i]
                                    -A[j,k]*A[k,i];
          A[j,i] := A[j,i] / A[j,j];
        end;
    end;

(* Marcha inversa. Solución de LY = b *)

  For i:=1 to Orden do
    begin
      Y[i]:=B[i];
      for k:=1 to i-1 do Y[i]:=Y[i] - A[i,k]*Y[k];
      Y[i]:=Y[i] / A[i,i];
    end;

(* Marcha inversa. Solución de UX = Y *)

  For i:=Orden downto 1 do
    begin
      X[i]:=Y[i];
      for k:=Orden downto i+1 do X[i]:=X[i]
                                    - A[i,k]*X[k];
    end;
end;


(*=====================================================*)
(*====FIN RESOLUCION DE UN SISTEMA DE ECUACIONES====*)
(*=====================================================*)


(*=====================================================*)
(*====== RESOLUCION DEL CIRCUITO EN CONTINUA  ======*)
(*=====================================================*)
```

```
Const

   Ids         = 1E-9;
             (* Corriente de saturación de los diodos *)
   Max_Diodos = 20;
                 (* Número máximo de diodos en la red *)

   VMax        = 2; (* Valor máximo tensión en diodos *)
   Vmin        = -1;(* Valor mínimo tensión en diodos *)

Type
                 (* Descripción diodos en el circuito *)
   Diodos = Array[1..Max_Diodos] of
     record
       Na,                            (* Nodo del ánodo *)
       Nc: integer;                  (* Nodo del cátodo *)
       Va: Real;   (* Tensión en la iteración actual *)
     end;


Function Id(V: real): real;
          (* Evaluación de la corriente por el diodo *)
Begin
   If V>Vmax then V:=Vmax;
             (* Límite máximo para evitar el overflow *)
   If V<Vmin
     then Id := - Ids
            (* Límite mínimo para evitar el underflow *)
     else Id:= Ids * ( exp ( V / 0.025 ) - 1);
End;

Function DId(V0: real): real;
      (* Evaluación de la derivada del diodo en V=V0 *)
Begin
   If V0>Vmax then V0:=Vmax;
   If V0<Vmin

     then V0 := Vmin;
           (* Límite mínimo para evitar el underflow *)
  DId:= Ids *  exp ( V0 / 0.025 ) /0.025;
End;


(* Procedimiento para la entrada de los diodos *)

Procedure Entrada_Diodos(var D: Diodos;
                         var ND: Integer);
Var
  i : integer;
```

```
Begin

  For i:=1 to ND do
    begin
      WriteLn;
      WriteLn('Diodo ',i,':');
      Write('Nodo del ánodo......................>');
      ReadLn(D[i].Na);
      Write('Nodo del cátodo.....................>');
      ReadLn(D[i].Nc);
      repeat
        Write('Tensión primera aproximación........>');
        ReadLn(D[i].Va);
      until Abs(D[i].Va) > 1E-10;
    end;
end;

(* Procedimiento para presentar los resultados *)

Procedure Presenta_Solucion (var V: Vector;
                                 N: Integer);
var
  i: integer;
begin
  For i:= 1 to N do
    Writeln('V',i,' = ',V[i]);
end;


(* Procedimiento que realiza el análisis del circuito
   no lineal. Parámetros:
   - Y      : Matriz de admitancias reducida.
   - I      : Vector de generadores independientes.
   - NN     : Número de nodos.
   - D      : Matriz de descripción de los diodos.
   - ND     : Número de Diodos.
   - E      : Error máximo permisible.
   - MI     : Número máximo de iteraciones. *)

Procedure Analisis(var  Y: Sistema; var G:Vector;
                   var NN: Integer; var D: Diodos;
                       ND:Integer;      E: real;

                       MI: Integer);

Var
(* Matriz real con los coeficientes del sistema de
   ecuaciones *)
   A     : Sistema;
```

```
   B,                        (* Términos independientes *)
   Vi    : Vector;              (* Solución iteración i *)
   NIt   : Integer;          (* Número iteración actual *)
   Final : Boolean;    (* Indica fin de la iteración *)
   Iter  : Boolean;
   i,j,k : integer;
   F,DF  : Real;                (* Variables auxiliares *)

Begin

  Final := False;
  NIt   := 1;

  If ND = 0

(* Si no hay diodos el circuito es lineal *)

    then begin

(* Se copia el sistema de partida *)

      A:=Y;
      B:=G;                  (* Términos independientes *)

(* Resolución del sistema en la presente iteración *)

      Resuelve_sistema(A,B,NN,Vi);

(* Se presentan los resultados *)

      Writeln;
      Writeln('Solución del circuito:');
      Presenta_Solucion (Vi,NN);

    end    (* Fin de tratamiento del análsis lineal *)

    else begin
            (* Hay diodos. El análisis es no lineal *)

      while ((NIt<=MI) and (not Final)) do
        begin

(* Se copia la parte real del sistema de partida *)

          A:=Y;
          B:=G;              (* Términos independientes *)

(* Se incorporan a A los efectos de los diodos *)
```

```
        For i:=1 to ND do
          with D[i] do
            begin
              F  :=  Id (Va);
                   (* Corriente en esta iteración *)
              DF := DId (Va);
                      (* Derivada de la corriente *)

              A[Na,Na] := A[Na,Na] + DF;
                       (* Resistencia Equivalente *)
              A[Nc,Nc] := A[Nc,Nc] + DF;
              A[Na,Nc] := A[Na,Nc] - DF;
              A[Nc,Na] := A[Nc,Na] - DF;

              B[Nc] := B[Nc]+F - DF * Va;
            (* Generador de corriente equivalente *)
              B[Na] := B[Na]-F + DF * Va;
            end;

(* Resolución del sistema en la presente iteración *)

        Resuelve_sistema(A,B,NN,Vi);

(* Se presentan los resultados *)

        Writeln;
        Writeln('Solución en la iteración ',NIt,':');
        Presenta_Solucion (Vi,NN);

(* Se actualizan las tensiones en diodos y se
   comprueba si se cumple el márgen de error *)

        Final:=True;
        For i:= 1 to ND do
          with D[i] do
            begin
              F:=Vi[Na]-Vi[Nc];
                (* Nuevo valor tensión en diodo i *)
              If Abs(Va-F) > E
                then Final:=False;
             (* Error diodo i mayor del permitido *)
              Va:= F;
            end;
        NIt:=NIt + 1;
      end;   (* Del WHILE. Fin de las iteraciones *)

(* Análisis de las causas de fin de las iteraciones *)
```

```
      If Final
        then
          Writeln('El sistema converge')
        else
          Writeln('Tras las iteraciones fijadas el',
                  ' sistema no converge');
    end;
 (* Del ELSE. Fin de tratamiento análisis no lineal *)

end;


(*=====================================================*)
(*= FIN DE LA RESOLUCION DEL CIRCUITO EN CONTINUA   =*)
(*=====================================================*)


(* Variables globales *)

Var

(* Matriz de admitancias del circuito *)

  Yn          : Sistema;

(* Matriz de generadores equivalentes *)

  Ieq         : Vector;

(* Número de nodos del circuito *)

  NNodos      : Integer;

(* Variables auxiliares *)

  i,j,k       : Integer;

(* Matriz con la información de los diodos *)

  Matriz_Diodos : Diodos;

(* Número de diodos del circuito *)

  Numero_Diodos : Integer;

(* Error máximo permisible en el resultado final *)

  Error : Real;
```

```
(* Número máximo de iteraciones a realizar *)

  Max_Iteraciones : Integer;

(*=====================================================*)
(*=RUTINAS AUXILIARES DE VISUALIZACION EN CONTINUA =*)
(*=====================================================*)

(* Procedimiento para visualizar la matriz de
   admitancias *)

Procedure Ver_Matriz_Admitancias(var Y: Sistema);

var i,j: integer;

begin
  Writeln;
  Writeln('Matriz de admitancias del circuito');
  Writeln('----------------------------------');
  Writeln;
  for i:=1 to NNodos do
    for j:=1 to NNodos do
        WriteLn('Y(',i,',',j,') = ',Y[i,j]);
end;

(* Procedimiento para visualizar la matriz de
   generadores independientes *)

Procedure Ver_Generadores( var I: Vector);

var j:integer;

begin
  Writeln;
  Writeln('Matriz de generadores independientes');
  Writeln('------------------------------------');
  Writeln;
  for j:=1 to NNodos do
    WriteLn('I(',j,') = ',I[j]);
end;

(*=====================================================
(*= FIN DE LAS RUTINAS AUXILIARES DE VISUALIZACION =
(*=====================================================

(* Programa Principal *)

BEGIN

(* Borrado de la matriz de admitancias *)
```

```
   for i:=0 to Max_Nodos do
     for j:=0 to Max_Nodos do
       Yn[i,j]:=0;

 (* Borrado matriz de generadores independientes *)

   for i:=0 to Max_Nodos do
     Ieq[i]:=0;

 (* Petición del número de nodos, comprobando si
    rebasa el máximo permitido *)

   repeat
     WriteLn('Introduzca el número de nodos del');
     Write(' circuito (Sin contar el de masa)...>');
    ReadLn(NNodos);
    If NNodos > Max_Nodos
      then writeln ('No es posible procesar tantos',
                    ' nodos con este programa');
    until NNodos <= Max_Nodos;

  Writeln;
  Writeln('ATENCION: Ponga su teclado en mayúsculas');
  Writeln;

(* Entrada de los componentes lineales del circuito *)

  Writeln('Introduzca los elementos lineales:');
  Writeln;

  Entrada_del_Circuito(Yn,Ieq);

  Ver_Matriz_Admitancias (Yn);

  Ver_Generadores (Ieq);

(* Entrada de los diodos *)

  WriteLn;
  repeat
    Write('Número de diodos en el circuito......>');
    ReadLn(Numero_Diodos);
  until Numero_Diodos <= Max_Diodos;

  If Numero_Diodos <> 0
    then begin
```

```
        Entrada_Diodos (Matriz_Diodos,Numero_Diodos);
        WriteLn;

        Write('Error máximo permisible.............>');
        ReadLn(Error);

        Write('Número máximo de iteraciones........>');
        ReadLn(Max_Iteraciones);

      end;

  (* Análisis del circuito *)

    Analisis(Yn,Ieq,NNodos,Matriz_Diodos,
             Numero_Diodos,Error,Max_Iteraciones);

  END.
```



Ejemplos

Para ilustrar el uso de este programa se van a resolver dos circuitos.

El primero es un simple circuito resistivo, conocido como atenuador en T puenteada.

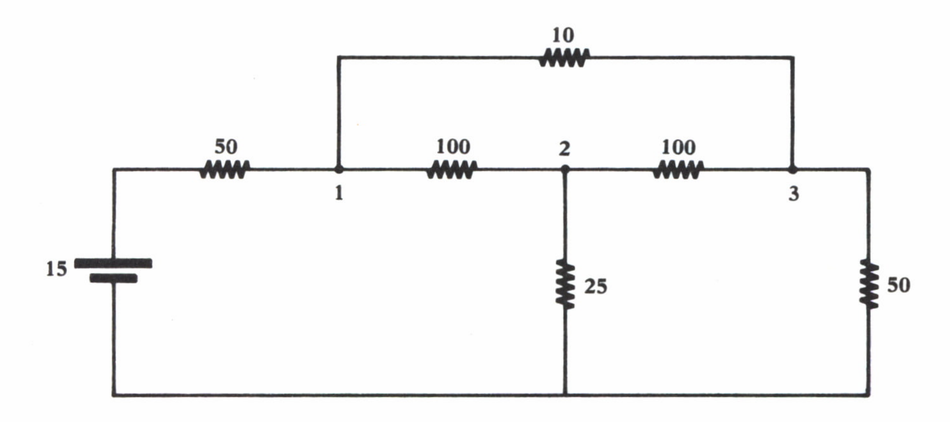


Fig. 35. Atenuador en T puenteada.

A continuación se puede ver el proceso completo de ejecución del programa con los resultados obtenidos:

```
Introduzca el número de nodos del
 circuito (Sin contar el de masa)...>3

ATENCION: Ponga su teclado en mayúsculas

Introduzca los elementos lineales:

Tipos posibles de componentes:
   - R: Resistencia.
   - V: Generador independiente de tensión.
   - I: Generador independiente de corriente.
   - Q: Fin del circuito

Introduzca el tipo de componente.....>V
Datos del generador:
Terminal negativo conectado al nudo..>0
Terminal positivo conectado al nudo..>1
Valor del generador..................>15
Valor del componente en serie........>50

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>1
Terminal negativo conectado al nodo..>2
Valor del componente.................>100
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>3
Valor del componente.................>100
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>0
Valor del componente.................>25
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>1
Terminal negativo conectado al nodo..>3
Valor del componente.................>10
Controla algún generador?............>N
```

```
Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>3
Terminal negativo conectado al nodo..>0
Valor del componente.................>50
Controla algún generador?............>N
Introduzca el tipo de componente.....>Q

Matriz de admitancias del circuito
----------------------------------

Y(1,1) =   1.3000000000E-01
Y(1,2) =  -1.0000000000E-02
Y(1,3) =  -1.0000000000E-01
Y(2,1) =  -1.0000000000E-02
Y(2,2) =   6.0000000000E-02
Y(2,3) =  -1.0000000000E-02
Y(3,1) =  -1.0000000000E-01
Y(3,2) =  -1.0000000000E-02
Y(3,3) =   1.3000000000E-01

Matriz de generadores independientes
------------------------------------

I(1) =   3.0000000000E-01
I(2) =   0.0000000000E+00
I(3) =   0.0000000000E+00

Número de diodos en el circuito......>0

Solución del circuito:
V1 =   6.2771739130E+00
V2 =   1.8750000000E+00
V3 =   4.9728260869E+00
```

El siguiente ejemplo corresponde a un circuito no lineal con dos diodos como el de la figura siguiente:

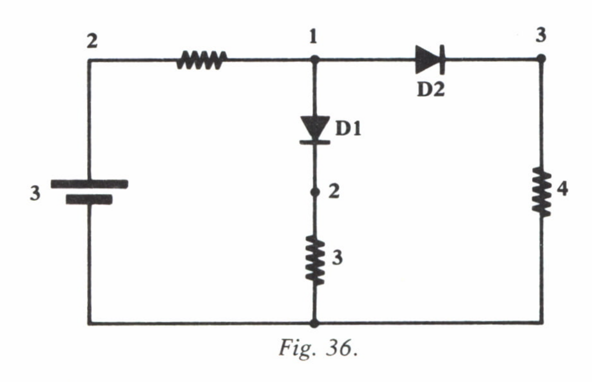


Fig. 36.

La ejecución del programa es como sigue:

```
Introduzca el número de nodos del
 circuito (Sin contar el de masa)...>3

ATENCION: Ponga su teclado en mayúsculas

Introduzca los elementos lineales:

Tipos posibles de componentes:
   - R: Resistencia.
   - V: Generador independiente de tensión.
   - I: Generador independiente de corriente.
   - Q: Fin del circuito

Introduzca el tipo de componente.....>V
Datos del generador:
Terminal negativo conectado al nudo..>0
Terminal positivo conectado al nudo..>1
Valor del generador..................>3
Valor del componente en serie........>2

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>0
Valor del componente.................>3
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>3
Terminal negativo conectado al nodo..>0
Valor del componente.................>4
Controla algún generador?............>N

Introduzca el tipo de componente.....>Q


Matriz de admitancias del circuito
----------------------------------


Y(1,1) =   5.0000000000E-01
Y(1,2) =   0.0000000000E+00
Y(1,3) =   0.0000000000E+00
Y(2,1) =   0.0000000000E+00
Y(2,2) =   3.3333333333E-01
Y(2,3) =   0.0000000000E+00
Y(3,1) =   0.0000000000E+00
Y(3,2) =   0.0000000000E+00
Y(3,3) =   2.5000000000E-01
```

```
Matriz de generadores independientes
------------------------------------

I(1) =   1.5000000000E+00
I(2) =   0.0000000000E+00
I(3) =   0.0000000000E+00

Número de diodos en el circuito......>2

Diodo 1:
Nodo del ánodo.......................>1
Nodo del cátodo......................>2
Tensión primera aproximación.........>0.5

Diodo 2:
Nodo del ánodo.......................>1
Nodo del cátodo......................>3
Tensión primera aproximación.........>0.5

Error máximo permisible..............>0.001
Número máximo de iteraciones.........>20

Solución en la iteración 1:
V1 =   1.6499377684E+00
V2 =   1.1550974902E+00
V3 =   1.1599944758E+00

Solución en la iteración 2:
V1 =   1.6491434466E+00
V2 =   1.1549186568E+00
V3 =   1.1618215642E+00

Solución en la iteración 3:
V1 =   1.6491074077E+00
V2 =   1.1548909201E+00
V3 =   1.1619306242E+00
El sistema converge
```

En este caso particular la convergencia es muy rápida. Si se hubieran tomado otras condiciones iniciales, el sistema podía haber resultado divergente o bien oscilar en torno a unos valores arbitrarios.

ANALISIS DE CIRCUITOS EN ALTERNA 4



L análisis en alterna es el más sencillo de los tres que componen el proceso completo. En él empleamos las impedancias de los componentes para obtener sus relaciones corriente-tensión, como se vio en el capítulo «Impedancias», en la página 20.

El proceso que debemos seguir es el siguiente:

1. Sustituir los componentes no lineales que existan por sus circuitos equivalentes para pequeña señal. Para ello deberemos conocer normalmente los resultados del análisis en continua, ya que algunos de los parámetros de este circuito dependen de aquéllos.

Con esto habremos llegado a un circuito con componentes exclusivamente lineales.

2. Plantear las ecuaciones del circuito por el método del análisis nodal.
3. Resolver el sistema formado.

El primer apartado se va a estudiar en detalle en este capítulo. El segundo ya se ha desarrollado con anterioridad, y para el tercero se puede consultar el apéndice «Resolución de sistemas de ecuaciones», en la página 134, donde se trata uno de los posibles métodos de resolución junto con un par de ejemplos. Para un estudio más detallado sobre la resolución de sistemas de ecuaciones, el lector hará bien en consultar la bibliografía básica que encontrará al final del libro, ya que este tema se sale fuera de la extensión del presente.

Del mismo modo que en el capítulo anterior, se incluye en éste un programa para ilustrar los métodos matemáticos presentados. El programa es también completo, esto es, lleva incluidas todas las rutinas necesarias para su correcto funcionamiento. El lector que no esté interesado en la justifi-

cación matemática del procedimiento a seguir puede acudir directamente a dicho programa, ya que los comentarios que en él se incluyen pueden resultar casi suficientes para comprender su forma de trabajo.



MODELO DE PEQUEÑA SEÑAL PARA EL TRANSISTOR BIPOLAR

Llamamos a este modelo «de pequeña señal» porque, para su uso, es necesario que la señal alterna que se aplica al transistor no sea de gran amplitud, para poder aproximar la respuesta del transistor en esa pequeña zona por una línea recta.

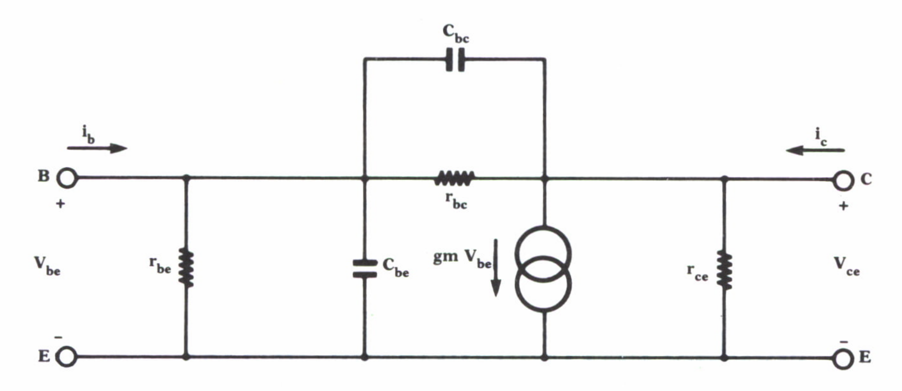


Fig. 37. Modelo del transistor bipolar en pequeña señal.

Normalmente se trabaja con transistores polarizados en zona activa, esto es, con la unión base-emisor en polarización directa y la unión base-colector en polarización inversa. El modelo de pequeña señal exige que la exploración de la señal alterna que se superpone a la continua sea tal que no lleve nunca el transistor al corte (ambas uniones polarizadas en inverso) ni a la saturación (ambas uniones bajo polarización directa).

El circuito al que nos referimos se halla en la figura 37. En él se han despreciado algunos efectos de poca importancia para resaltar los fundamentales. Los parámetros que necesitaremos conocer son:

- hfe: ganancia en corriente. Aunque no aparece en el circuito, la emplearemos en el cálculo de otros componentes. Su valor viene dado por:

$$hfe = \frac{Ic}{Ib}$$

Siendo Ic e Ib las del punto de polarización.

• rbe: resistencia base-emisor. Viene dada por la expresión:

$$rbe = \frac{Vt}{Ib} \quad ,, \quad Vt = \frac{K\,T}{q} = 0{,}025 \text{ a temperatura ambiente.}$$

• Cbe, Cbc: Capacidades base-emisor y base-colector. Son de un valor muy pequeño (algunos picofaradios) y se pueden tomar de las características del transistor dadas por el fabricante sin cometer un gran error.

• rbc: resistencia base-colector. Se obtiene también de las características dadas por el fabricante, considerando que

$$rbc = \frac{rbe}{hre} \quad ,, \text{ hre: Ganancia de corriente inversa en emisor común.}$$

Su valor es muy alto (4 MΩ), por lo que normalmente se elimina.

• gm: Transconductancia de salida. Su valor viene dado por:

$$gm = \frac{hfe}{Rbe}$$

• rce: resistencia de salida. Se obtiene a partir de las características del fabricante, ya que rce = 1/hoe.



MODELO DE PEQUEÑA SEÑAL DE LOS TRANSISTORES DE EFECTO DE CAMPO

Las ecuaciones que representan el comportamiento de los transistores FET de unión y de estructura MOS (MOSFET), en régimen de pequeña señal, son del mismo tipo. Por tanto, los podremos sustituir por el mismo circuito equivalente para esta fase de cálculo.

La figura 38 presenta dicho circuito.

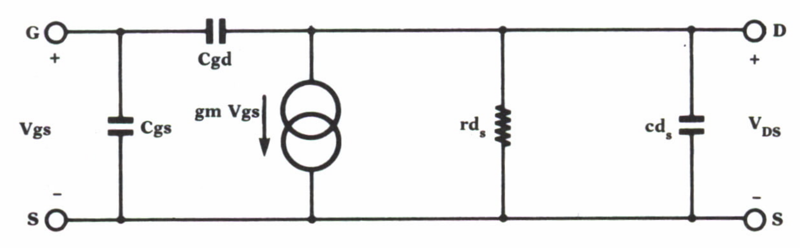


Fig. 38. Modelo de los transistores de efecto de campo.

Sus parámetros se pueden derivar a partir de las hojas de características del fabricante. La variedad de notaciones con que se les denomina es muy grande. Aquí se presentan algunas de las más corrientes:

gm, gfs, yfs: Transconductancia directa.

gm0: Valor de gm cuando Vgs = 0.

Vp, Vgs(off): Tensión de estrangulamiento o de «Pinch-off». Es el valor de la tensión puerta-fuente por encima del cual la corriente drenador-fuente no aumenta.

Idss: Corriente de drenador para Vgs = 0.

Ciss, Cgss: Capacidad de entrada en configuración de fuente común, con la salida en cortocircuito.

Crss: Capacidad inversa de transferencia, con la entrada cortocircuitada.

Coss: Capacidad de salida en fuente común, con la entrada cortocircuitada.

gd, yos, gos: Conductancia de salida en fuente común.

Las ecuaciones que podemos aplicar son:

$$gm = gm0\ (1 - \frac{Vgs}{Vp})$$

$$Vp = \frac{2\ Idss}{gm0}$$

Cds = Coss – Crss Normalmente es muy pequeña (<1pf) y se desprecia.

Cgd = Crss o bien, aproximadamente, $Cgd = \frac{1}{\sqrt{-Vgd}}$

Cgs = Ciss – Crss o bien, aproximadamente, $Cgs = \frac{1}{\sqrt{-Vgs}}$

Es preciso señalar, sin embargo, que los datos del fabricante suelen tener una dispersión tal que hacen que el modelo sea solamente aproximado.



MODELO DEL AMPLIFICADOR OPERACIONAL

El amplificador operacional es un circuito integrado lineal, comúnmente empleado hoy en día como un elemento más de circuito, sin ocuparse de su estructura interna. Es por esto por lo que resulta interesante disponer de un modelo de él, que nos permita realizar el análisis de los circuitos en los que intervenga de una forma sencilla.

La representación simbólica del amplificador operacional se ofrece en la figura 39.

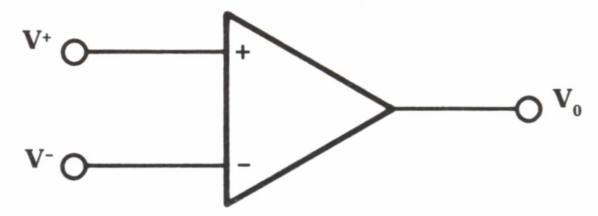


Fig. 39. Diagrama del amplificador operacional.

Cuando se analizan circuitos con amplificadores operacionales por los métodos tradicionales (es decir, con lápiz y papel), se realizan una serie de simplificaciones para facilitar el cálculo. En el análisis por ordenador no es necesario realizarlas, con lo que conseguimos un estudio más próximo a la realidad. En el siguiente cuadro se van a estudiar las principales diferencias entre el análisis simplificado y un modelo más real:

Parámetro	**Simb.**	**Mod. simplif.**	**Mod. real**
Imp. de entrada	Zi	Infinita	Finita
Imp. de salida	Zo	Nula	No nula
Gan. en tensión	Av	Infinita	Finita
Ancho de banda	BW	Infinito	Finito

Con estas consideraciones podemos establecer el siguiente modelo del amplificador operacional para el análisis en el dominio de la frecuencia:

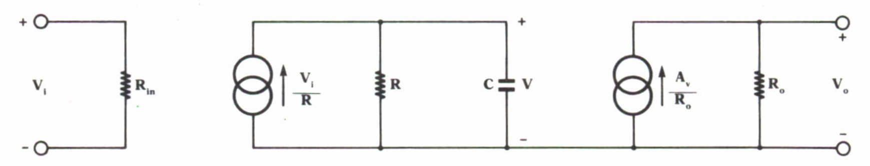


Fig. 40. Modelo de un amplificador operacional real en pequeña señal.

Los parámetros Av, Rin y Ro se obtienen directamente de las características dadas por el fabricante. En cuanto a R y C se fijan a partir de la frecuencia dada por el fabricante para el polo dominante en circuito abierto, que denominaremos f0, según la relación:

$$R.C = \frac{1}{2 \, . \, \pi \, . \, f0}$$

Valores aproximados de estos parámetros son:

Ri = 2 MΩ Av = 200.000
Ro = 75 Ω f0 = 10 Hz.

Con esto hemos logrado simplificar un circuito complejo, que ocuparía un volumen de memoria en nuestro ordenador enorme y un tiempo de cálculo bastante extenso, por otro más simple y que, con pequeñas restricciones, se comporta prácticamente igual para señales alternas de baja amplitud. Esta técnica recibe el nombre de *macromodelado*, y es extensible a cualquier otro tipo de circuito de uso general.

El macromodelo que realicemos puede ser tanto más general como nuestro análisis lo requiera. En cualquier caso, deberá reflejar los principales fenómenos que ocurran en la realidad bajo las condiciones de trabajo. En el caso del amplificador operacional, por ejemplo:

- Para el análisis en continua, se incluirían:
  - Corrientes de polarización.
  - Tensiones de offset.
  - Característica de transferencia estática.
  - Derivas de los parámetros.
- Para el análisis de pequeña señal más detallado:
  - Capacidad de entrada.
  - Factor de rechazo del modo común (CMMR).
  - Figura de ruido.
- Para el análisis en gran señal:
  - Slew-rate (velocidad de subida).
- Para el análisis transitorio, en función de la respuesta a un escalón de tensión en su entrada:
  - Tiempo de subida.
  - Tiempo de asentamiento.
  - Tiempo de adquisición.

Que son de interés en muchas aplicaciones (comparadores, muestreo-retención).



EJEMPLO DE PROGRAMACION

Continuando en la línea de los ejemplos hasta ahora introducidos, se ofrece este nuevo programa. En él se emplean las rutinas de entrada del capítulo «Fundamentos del diseño de circuitos asistido por ordenador», en

la página 13, del mismo modo que las rutinas de resolución de un sistema de ecuaciones complejas presentadas en el «Resolución de sistemas de ecuaciones», en la página 134.

Como se puede apreciar, el programa no tiene la complejidad del caso de resolución de circuitos no lineales en continua. Por el contrario, los resultados que de él se obtienen no dependen de parámetros a fijar de forma aproximada «a priori», con lo que su fiabilidad es mucho mayor.

De igual forma que en anteriores programas, los frecuentes comentarios deben ser suficientes para comprender el funcionamiento de las rutinas que aquí nos ocupan. En este caso, su brevedad hace innecesaria cualquier otra posible explicación. Pasemos, pues, a verlo:

```
Program Analisis_en_alterna;

(* Este programa realiza el análisis en alterna de
   un circuito para un margen de frecuencias
   especificado *)


(*=====================================================*)
(*= PROCEDIMIENTOS PARA LA  ENTRADA  DEL  CIRCUITO =*)
(*=====================================================*)
(* Estos procedimientos son los mismos que los
   correspondientes del programa Entrada_de_Circuito,
   ya estudiado anteriormente, y por eso se presentan
   aquí sin comentarios. *)
Const
  Max_Nodos = 20;
Type
  Circuito    = Array [0..Max_Nodos,0..Max_Nodos,
                       -1..1] of real;
  Generadores = Array[0..Max_Nodos] of real;
Procedure Entrada_del_Circuito ( var Y: Circuito;
                                 var I: Generadores);
var
  Componente       : Char;
  N1,
  N2,
  NC1,
  NC2,
  TipoG            : Integer;
  ValorComponente,
  ValorGenerador,
  GGenerador       : Real;
  Error            : Boolean;
procedure Gen_Tension(var N1,N2: integer;
                      var valor,G_serie : real);
```

```
begin
  writeln('Datos del generador:');
  write('Terminal negativo conectado al nudo..>');
  readln(N1);
  write('Terminal positivo conectado al nudo..>');
  readln(N2);
  write('Valor del generador..................>');
  readln(valor);
  write('Valor del componente en serie........>');
  readln(G_serie);
  G_Serie := 1/G_Serie;
  Valor:=Valor * G_Serie;
end;
procedure Gen_Corriente(var N1,N2: integer;
                        var valor : real);
begin
  writeln('Datos del generador:');
  write('Terminal origen conectado al nudo....>');
  readln(N1);
  write('Terminal destino conectado al nudo...>');
  readln(N2);
  write('ValorProgram Analisis_en_alterna;

(* Este programa realiza el análisis en alterna de
   un circuito para un margen de frecuencias
   especificado *)


(*=====================================================*)
(*= PROCEDIMIENTOS PARA LA  ENTRADA  DEL  CIRCUITO =*)
(*=====================================================*)
(* Estos procedimientos son los mismos que los
   correspondientes del programa Entrada_de_Circuito,
   ya estudiado anteriormente, y por eso se presentan
   aquí sin comentarios. *)
Const
  Max_Nodos = 20;
Type
  Circuito    = Array [0..Max_Nodos,0..Max_Nodos,
                      -1..1] of real;
  Generadores = Array[0..Max_Nodos] of real;
Procedure Entrada_del_Circuito ( var Y: Circuito;
                                 var I: Generadores);
var
  Componente       : Char;
  N1,
  N2,
  NC1,
  NC2,
```

```
   TipoG             : Integer;
   ValorComponente,
   ValorGenerador,
   GGenerador        : Real;
   Error             : Boolean;
procedure Gen_Tension(var N1,N2: integer;
                      var valor,G_serie : real);
begin
   writeln('Datos del generador:');
   write('Terminal negativo conectado al nudo..>');
   readln(N1);
   write('Terminal positivo conectado al nudo..>');
   readln(N2);
   write('Valor del generador..................>');
   readln(valor);
   write('Valor del componente en serie........>');
   readln(G_serie);
   G_Serie := 1/G_Serie;
   Valor:=Valor * G_Serie;
end;
procedure Gen_Corriente(var N1,N2: integer;
                        var valor : real);
begin
   writeln('Datos del generador:');
   write('Terminal origen conectado al nudo....>');
   readln(N1);
   write('Terminal destino conectado al nudo...>');
   readln(N2);
   write('Valor del generador..................>');
   readln(valor);
end;
procedure Entrada_Componente (C: integer);
var
   Control          : char;
begin
   TipoG:=0;
   Error:=False;
   writeln('Datos del componente:');
   write('Terminal positivo conectado al nodo..>');
   readln(N1);
   write('Terminal negativo conectado al nodo..>');
   readln(N2);
   write('Valor del componente.................>');
   readln(ValorComponente);
   if ((c= 0) or (c = -1)) then ValorComponente:=
                                1/ValorComponente;
   Control:=' ';
   while not (Control in ['N','S']) do
      begin
```

```
    write('Controla algún generador?...........>');
    readln(Control);
  end;
if Control ='S' then
  begin
    write('Variable que lo controla (I/V)......>');
    readln(Control);
    case control of
      'V': TipoG:=1;
      'I': TipoG:=2;
      else Error:=True;
    end;
    write('Tipo de generador (I/V).............>');
    readln(Control);
    case Control of
      'V': begin
             TipoG:=TipoG + 10;
             Gen_Tension(NC1, NC2, ValorGenerador,
                                   GGenerador);
           end;
      'I': begin
             TipoG:=TipoG + 20;
             Gen_Corriente(NC1, NC2,ValorGenerador);
           end;
      else
           Error:=True;
    end;
  end;

If not Error then
  begin
    Y[N1,N1,C]:=Y[N1,N1,C]+ValorComponente;
    Y[N2,N2,C]:=Y[N2,N2,C]+ValorComponente;
    Y[N1,N2,C]:=Y[N1,N2,C]-ValorComponente;
    Y[N2,N1,C]:=Y[N2,N1,C]-ValorComponente;
    if TipoG div 10 = 1 then
      begin
        Y[NC1,NC1,0]:=Y[NC1,NC1,0]+GGenerador;
        Y[NC2,NC2,0]:=Y[NC2,NC2,0]+GGenerador;
        Y[NC1,NC2,0]:=Y[NC1,NC2,0]-GGenerador;
        Y[NC2,NC1,0]:=Y[NC2,NC1,0]-GGenerador;
      end;
    if TipoG mod 10 = 1 then
      begin
        Y[NC1,N1,0]:=Y[NC1,N1,0]+ValorGenerador;
        Y[NC2,N2,0]:=Y[NC2,N2,0]+ValorGenerador;
        Y[NC1,N2,0]:=Y[NC1,N2,0]-ValorGenerador;
        Y[NC2,N1,0]:=Y[NC2,N1,0]-ValorGenerador;
      end;
```

```
      if TipoG mod 10 = 2 then
        begin
          Y[NC1,N1,0]:=Y[NC1,N1,0]+ValorGenerador*
                                    ValorComponente;
          Y[NC2,N2,0]:=Y[NC2,N2,0]+ValorGenerador*
                                    ValorComponente;
          Y[NC1,N2,0]:=Y[NC1,N2,0]-ValorGenerador*
                                    ValorComponente;
          Y[NC2,N1,0]:=Y[NC2,N1,0]-ValorGenerador*
                                    ValorComponente;
        end;
    end
    else writeln('Entrada equivocada. ',
                 'Repítala, por favor.');
end;
begin
  writeln('Tipos posibles de componentes:');
  writeln('  - R: Resistencia.');
  writeln('  - L: Bobina.');
  writeln('  - C: Condensador.');
  writeln('  - V: Generador indep. de tensión.');
  writeln('  - I: Generador indep. de corriente.');
  writeln('  - Q: Fin del circuito');

  repeat
    WriteLn;
    Write('Introduzca el tipo de componente.....>');
    ReadLn(Componente);
    Case Componente of
      'R' : Entrada_Componente(0);
      'L' : Entrada_Componente(-1);
      'C' : Entrada_Componente(1);
      'V' : begin
              Gen_Tension(NC1, NC2, ValorGenerador,
                                    GGenerador);
              Y[NC1,NC1,0]:=Y[NC1,NC1,0]+GGenerador;
              Y[NC2,NC2,0]:=Y[NC2,NC2,0]+GGenerador;
              Y[NC1,NC2,0]:=Y[NC1,NC2,0]-GGenerador;
              Y[NC2,NC1,0]:=Y[NC2,NC1,0]-GGenerador;
              I[NC1]:=I[NC1]-ValorGenerador;
              I[NC2]:=I[NC2]+ValorGenerador;
            end;
      'I' : begin
              Gen_Corriente(NC1, NC2, ValorGenerador);
              I[NC1]:=I[NC1]-ValorGenerador;
              I[NC2]:=I[NC2]+ValorGenerador;
            end;
      end;
```

```
    until Componente='Q'
end;

(*=====================================================*)
(*FIN DE LOS PROCEDIMIENTOS DE ENTRADA DEL CIRCUITO *)
(*=====================================================*)

(*=====================================================*)
(*=RESOLUCION DE UN SISTEMA DE ECUACIONES COMPLEJAS=*)
(*=====================================================*)
(* Estos procedimientos son los mismos que los
   correspondientes del programa
   Sistema_de_Ecuaciones_Complejas, presentado en los
   apéndices, y por eso se presentan aquí sin
   comentarios. *)
Const
  Orden_Maximo = 20;
Type
  Complejo= array[1..2] of real;
  Sistema = array[1..Orden_Maximo,1..Orden_Maximo]
                  of Complejo;
  Vector  = array[1..Orden_Maximo] of complejo;
Procedure R_P(Var C:Complejo);
var
  R: real;
begin
  R:=Sqrt(Sqr(C[1])+Sqr(C[2]));
  if Abs(C[1]) < 1E-10
    then
      begin
        if C[2]>0
          then C[2]:=Pi/2
          else C[2]:=-Pi/2;
      end
    else
      begin
        if C[1]>0
          then C[2]:=ArcTan(C[2]/C[1])
          else C[2]:=ArcTan(C[2]/C[1]) + Pi;
      end;
  C[1]:=R;
end;
Procedure IguC(C1: Complejo;var Cr:Complejo);
var
  i:integer;
Begin
  For i:=1 to 2 do Cr[i]:=C1[i];
end;
```

```
Procedure SumC(C1,C2: Complejo;var Cr:Complejo);
var
  i:integer;
Begin
  For i:=1 to 2 do Cr[i]:=C1[i]+C2[i];
end;
Procedure ResC(C1,C2: Complejo;var Cr:Complejo);
var
  i:integer;
Begin
  For i:=1 to 2 do Cr[i]:=C1[i]-C2[i];
end;
Procedure MulC(C1,C2: Complejo;var Cr:Complejo);
var
  i:integer;
Begin
  Cr[1]:=C1[1]*C2[1]-C1[2]*C2[2];
  Cr[2]:=C1[1]*C2[2]+C1[2]*C2[1];
end;
Procedure DivC(C1,C2: Complejo;var Cr:Complejo);
var
  i:integer;
  D: real;
Begin
  D:=sqr(C2[1])+Sqr(C2[2]);
  Cr[1]:=(C1[1]*C2[1]+C1[2]*C2[2])/D;
  Cr[2]:=(-C1[1]*C2[2]+C1[2]*C2[1])/D;
end;
Procedure Resuelve_Sistema(var A: Sistema;
      var B: vector; Orden: Integer;var X:Vector);
var
  i,j,k   : integer;
  Y       : Vector;
  C       : Complejo;
Begin
  For j:=1 to Orden do
    Begin
      for i:=j to Orden do
        begin
          for k:=1 to j-1 do
            begin
              MulC(A[i,k],A[k,j],C);
              ResC(A[i,j],C,A[i,j]);
            end;
        end;
      for i:=j+1 to Orden do
        begin
          for k:=1 to j-1 do
```

```
            begin
              MulC(A[j,k],A[k,i],C);
              ResC(A[j,i],C,A[j,i]);
            end;
          DivC(A[j,i],A[j,j],A[j,i]);
          end;
      end;
   For i:=1 to Orden do
     begin
       IguC(B[i],Y[i]);
       for k:=1 to i-1 do
         begin
           MulC(A[i,k],Y[k],C);
           ResC(Y[i],C,Y[i]);
         end;
       DivC(Y[i],A[i,i],Y[i]);
     end;
   For i:=Orden downto 1 do
     begin
       IguC(Y[i],X[i]);
       for k:=Orden downto i+1 do
         begin
           MulC(A[i,k],X[k],C);
           ResC(X[i],C,X[i]);
         end;
     end;
end;

(*====================================================*)
(*====FIN RESOLUCION DE UN SISTEMA DE ECUACIONES====*)
(*====================================================*)

(*====================================================*)
(*==============  ANALSIS EN ALTERNA ===============*)
(*====================================================*)

(* Procedimiento para resolver el circuito en alterna
     a una frecuencia determinada. Parámetros de
     entrada:

    - Y   : Matriz de admitancias del circuito.
    - I   : Matriz de generadores de corriente.
    - N   : Numero de nodos.
    - F   : Frecuencia a la que se efectúa el análisis.
    - V   : Solución del circuito a la frecuencia dada
            (vector complejo).*)

Procedure Solucion_en_F(Var Y: circuito;
```

```
                        Var Ig:Generadores; N: integer;
                        F: real; Var V:Vector);

Var

  Ya : Sistema;
(* Matriz del sistema de ecuaciones a una frecuencia*)
  B  : Vector;            (* Términos independientes *)
  i,j: integer;

Begin
  F:=2*Pi*F;
  (* Cálculo de la pulsación w a partir de la frec. *)

(* A continuación se calcula la matriz de admitancias
   del sistema para la frecuencia de interés *)

  For i:=1 to N do
    begin
      For j:=1 to N do
        begin
          Ya[i,j][1]:=Y[i,j,0];
          Ya[i,j][2]:=Y[i,j,1] * F - Y[i,j,-1] / F ;
        end;
      B[i][1]:=Ig[i];
      B[i][2]:=0;
    end;

(* Se resuelve el sistema obtenido *)

  Resuelve_Sistema(Ya,B,N,V);
end;

(*=====================================================*)
(*==========  FIN DEL ANALISIS EN ALTERNA  =========*)
(*=====================================================*)

(* Variables globales *)

Var

  Yn         : Circuito;
               (* Matriz de admitancias del circuito *)
  Ieq        : Generadores;
               (* Matriz de generadores equivalentes *)
  NNodos     : Integer;
                     (* Número de nodos del circuito *)
  F    ,                    (* Frecuencia a analizar *)
  Fmax,              (* Frecuencia máxima a analizar *)
```

```
  dF        : Real;      (* Incremento de frecuencia *)
  Vk        : Vector;       (* Solución del circuito *)
  i,j,k     : Integer;        (* Variables auxiliares *)



(*==================================================*)
(*====== RUTINAS AUXILIARES DE VISUALIZACION =======*)
(*==================================================*)

Procedure Ver_Matriz_Admitancias(var Y: Circuito);

var i,j: integer;

begin
  Writeln;
  Writeln('Matriz de admitancias del circuito');
  Writeln('----------------------------------');
  Writeln;
  for i:=1 to NNodos do
    for j:=1 to NNodos do begin
      WriteLn('Y(',i,',',j,') = ',Y[i,j,-1],' 1/jw');
      WriteLn(' + ',Y[i,j,0],' + ',Y[i,j,1],' jw');
      end;
end;

Procedure Ver_Generadores( var I: Generadores);

var j:integer;

begin
  Writeln;
  Writeln('Matriz de generadores independientes');
  Writeln('------------------------------------');
  Writeln;
  for j:=1 to NNodos do
    WriteLn('I(',j,') = ',I[j]);
end;

(*==================================================*)
(*= FIN DE LAS RUTINAS AUXILIARES DE VISUALIZACION =*)
(*==================================================*)

(* Programa Principal *)

BEGIN

(* Borrado de la matriz de admitancias *)
```

```
  for i:=0 to Max_Nodos do
    for j:=0 to Max_Nodos do
      for k:=-1 to 1 do
        Yn[i,j,k]:=0;

(* Borrado matriz de generadores independientes *)

  for i:=0 to Max_Nodos do
    Ieq[i]:=0;

(* Petición del número de nodos, comprobando si
   rebasa el máximo permitido *)

  repeat
    WriteLn('Introduzca el número de nodos del');
    Write(' circuito (Sin contar el de masa)...>');
    ReadLn(NNodos);
    If NNodos > Max_Nodos
      then writeln ('No es posible procesar tantos',
                    ' nodos con este programa');
    until NNodos <= Max_Nodos;
  Writeln;
  Writeln('ATENCION: Ponga su teclado en mayúsculas');
  Writeln;

  Entrada_del_Circuito(Yn,Ieq);

  Ver_Matriz_Admitancias(Yn);

  Ver_Generadores(Ieq);

  WriteLn;
  WriteLn('ANALISIS EN ALTERNA DEL CIRCUITO');
  WriteLn('--------------------------------');
  WriteLn;

(* Se piden los límites de frecuencia para los que se
   debe realizar el análisis *)

  Write('Frecuencia mínima para analizar.....>');
  ReadLn(F);
  Write('Frecuencia máxima para analizar.....>');
  ReadLn(Fmax);
  Write('Factor incremento de frecuencias.....>');
  ReadLn(dF);


  While F <= Fmax do
    begin
```

```
    Solucion_en_F(Yn,Ieq,NNodos,F,Vk);
    Writeln;
    Writeln('Solución en F= ',F,' Hz.');
    for i:=1 to NNodos do
                    (* Se presentan de los resultados *)
      begin
        WriteLn('V',i,': ',Vk[i][1]:10,
                ' +j( ',Vk[i][2]:10,')');
                     (* Coordenadas rectangulares *)
        R_P(Vk[i]);
        Vk[i][2]:=Vk[i][2] * 180 / Pi;
                     (* Paso de radianes a grados *)
        Writeln('  ==> ',Vk[i][1]:10,
                ' * exp[j(',Vk[i][2]:10,')]');
                           (* Coordenadas polares *)
      end;

    F:=F*dF;       (* Para pasar de una frecuencia a
                      la siguiente del análisis
                      se multiplicará por dF *)
  end;
END.
```

Para ilustrar su funcionamiento se va a analizar una etapa amplificadora en emisor común con acoplo en alterna, cuyo esquema eléctrico es como sigue:

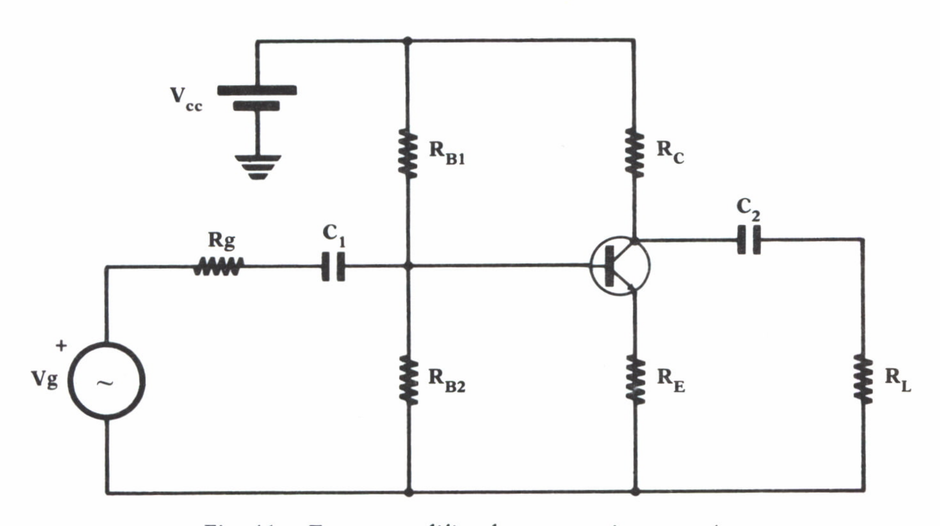


Fig. 41. Etapa amplificadora en emisor común.

El circuito equivalente, empleando el modelo del transistor para pequeña señal, tiene la forma:

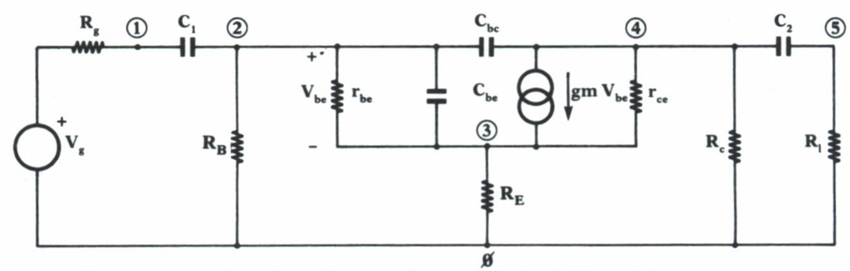


Fig. 42. Circuito equivalente en pequeña señal.

En el que se ha puesto ya la asignación de nodos elegida para realizar el análisis.

Los valores de los componentes son:

Vg = 1	RE = 100 Ω	C1 = 1 μF
Rg = 50 Ω	rbe = 1 KΩ	C2 = 1 μF
RB = RB1 // RB2 = 25 KΩ	rce = 35 KΩ	gm = 0,36
Rc = 1 KΩ	Cbc = 10 pF	
R1 = 1 KΩ	Cbe = 25 pF	

Los valores de los parámetros del transistor se obtienen de las condiciones fijadas por su punto de polarización, como se ha explicado en este capítulo.

Proceso y resultado del análisis vienen resumidos en el siguiente listado:

```
Introduzca el número de nodos del
 circuito (Sin contar el de masa)...>5

ATENCION: Ponga su teclado en mayúsculas

Tipos posibles de componentes:
  - R: Resistencia.
  - L: Bobina.
  - C: Condensador.
  - V: Generador indep. de tensión.
  - I: Generador indep. de corriente.
  - Q: Fin del circuito

Introduzca el tipo de componente.....>V
Datos del generador:
Terminal negativo conectado al nudo..>0
Terminal positivo conectado al nudo..>1
Valor del generador..................>1
Valor del componente en serie........>50
```

```
Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>1
Terminal negativo conectado al nodo..>2
Valor del componente.................>1E-6

Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>0
Valor del componente.................>25E3
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>3
Valor del componente.................>1E3
Controla algún generador?............>S
Variable que lo controla (I/V).......>V
Tipo de generador (I/V)..............>I
Datos del generador:
Terminal origen conectado al nudo....>4
Terminal destino conectado al nudo...>3
Valor del generador..................>0.36

Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>3
Valor del componente.................>25E-12
Controla algún generador?............>N
Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>2
Terminal negativo conectado al nodo..>4
Valor del componente.................>10E-12
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>3
Terminal negativo conectado al nodo..>0
Valor del componente.................>100
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>4
```

```
Terminal negativo conectado al nodo..>3
Valor del componente.................>35E3
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>4
Terminal negativo conectado al nodo..>0
Valor del componente.................>1E3
Controla algún generador?............>N

Introduzca el tipo de componente.....>C
Datos del componente:
Terminal positivo conectado al nodo..>4
Terminal negativo conectado al nodo..>5
Valor del componente.................>1E-6
Controla algún generador?............>N

Introduzca el tipo de componente.....>R
Datos del componente:
Terminal positivo conectado al nodo..>5
Terminal negativo conectado al nodo..>0
Valor del componente.................>1E3
Controla algún generador?............>N

Introduzca el tipo de componente.....>Q

Matriz de admitancias del circuito
----------------------------------

Y(1,1) =   0.0000000000E+00 1/jw
 +   2.0000000000E-02 +   1.0000000000E-06 jw
Y(1,2) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-06 jw
Y(1,3) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(1,4) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(1,5) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(2,1) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-06 jw
Y(2,2) =   0.0000000000E+00 1/jw
 +   1.0400000000E-03 +   1.0000350000E-06 jw
Y(2,3) =   0.0000000000E+00 1/jw
 +  -1.0000000000E-03 +  -2.5000000000E-11 jw
Y(2,4) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +  -1.0000000000E-11 jw
Y(2,5) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
Y(3,1) =   0.0000000000E+00 1/jw
 +   0.0000000000E+00 +   0.0000000000E+00 jw
```

```
Y(3,2) =    0.0000000000E+00 1/jw
 +   -3.6100000000E-01 +   -2.5000000000E-11 jw
Y(3,3) =    0.0000000000E+00 1/jw
 +    3.7102857143E-01 +    2.5000000000E-11 jw
Y(3,4) =    0.0000000000E+00 1/jw
 +   -2.8571428571E-05 +    0.0000000000E+00 jw
Y(3,5) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(4,1) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(4,2) =    0.0000000000E+00 1/jw
 +    3.6000000000E-01 +   -1.0000000000E-11 jw
Y(4,3) =    0.0000000000E+00 1/jw
 +   -3.6002857143E-01 +    0.0000000000E+00 jw
Y(4,4) =    0.0000000000E+00 1/jw
 +    1.0285714286E-03 +    1.0000100000E-06 jw
Y(4,5) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +   -1.0000000000E-06 jw
Y(5,1) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(5,2) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(5,3) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +    0.0000000000E+00 jw
Y(5,4) =    0.0000000000E+00 1/jw
 +    0.0000000000E+00 +   -1.0000000000E-06 jw
Y(5,5) =    0.0000000000E+00 1/jw
 +    1.0000000000E-03 +    1.0000000000E-06 jw

Matriz de generadores independientes
------------------------------------

I(1) =   2.0000000000E-02
I(2) =   0.0000000000E+00
I(3) =   0.0000000000E+00
I(4) =   0.0000000000E+00
I(5) =   0.0000000000E+00

ANALISIS EN ALTERNA DEL CIRCUITO
--------------------------------

Frecuencia mínima para analizar......>1
Frecuencia máxima para analizar......>10E6
Factor incremento de frecuencias.....>10

Solución en F=   1.0000000000E+00 Hz.
V1: 9.9997E-01 +j( -3.115E-04)
  ==> 9.9997E-01 * exp[j(-1.785E-02)]
V2: 8.5442E-03 +j( 9.1911E-02)
  ==> 9.2308E-02 * exp[j(8.4689E+01)]
V3: 8.3064E-03 +j( 8.9359E-02)
  ==> 8.9744E-02 * exp[j(8.4689E+01)]
```

```
V4: -8.842E-02 +j( -8.904E-01)
  ==> 8.9482E-01 * exp[j(2.6433E+02)]
V5: 5.5911E-03 +j( -5.907E-04)
  ==> 5.6222E-03 * exp[j(-6.031E+00)]

Solución en F=   1.0000000000E+01 Hz.
V1: 9.9843E-01 +j( -1.684E-03)
  ==> 9.9843E-01 * exp[j(-9.665E-02)]
V2: 4.6234E-01 +j( 4.9730E-01)
  ==> 6.7902E-01 * exp[j(4.7087E+01)]
V3: 4.4947E-01 +j( 4.8351E-01)
  ==> 6.6016E-01 * exp[j(4.7089E+01)]
V4: -4.745E+00 +j( -4.507E+00)
  ==> 6.5443E+00 * exp[j(2.2352E+02)]
V5: 2.6339E-01 +j( -3.147E-01)
  ==> 4.1038E-01 * exp[j(-5.007E+01)]

Solución en F=   1.0000000000E+02 Hz.
V1: 9.9667E-01 +j( -3.503E-04)
  ==> 9.9667E-01 * exp[j(-2.014E-02)]
V2: 9.8552E-01 +j( 1.0564E-01)
  ==> 9.9116E-01 * exp[j(6.1185E+00)]
V3: 9.5835E-01 +j( 1.0291E-01)
  ==> 9.6386E-01 * exp[j(6.1292E+00)]
V4: -6.881E+00 +j( 1.6159E+00)
  ==> 7.0680E+00 * exp[j(1.6678E+02)]
V5: -2.676E+00 +j( -2.642E+00)
  ==> 3.7603E+00 * exp[j(2.2464E+02)]
Solución en F=   1.0000000000E+03 Hz.
V1: 9.9664E-01 +j( -5.302E-05)
  ==> 9.9664E-01 * exp[j(-3.048E-03)]
V2: 9.9647E-01 +j( 1.0640E-02)
  ==> 9.9653E-01 * exp[j(6.1177E-01)]
V3: 9.6916E-01 +j( 1.0378E-02)
  ==> 9.6922E-01 * exp[j(6.1351E-01)]
V4: -4.867E+00 +j( 3.3021E-01)
  ==> 4.8778E+00 * exp[j(1.7612E+02)]
V5: -4.798E+00 +j( -4.334E-01)
  ==> 4.8172E+00 * exp[j(1.8516E+02)]

Solución en F=   1.0000000000E+04 Hz.
V1: 9.9664E-01 +j( -1.881E-04)
  ==> 9.9664E-01 * exp[j(-1.081E-02)]
V2: 9.9658E-01 +j( 8.8133E-04)
  ==> 9.9658E-01 * exp[j(5.0670E-02)]
V3: 9.6927E-01 +j( 8.6040E-04)
  ==> 9.6927E-01 * exp[j(5.0860E-02)]
V4: -4.833E+00 +j( 3.6016E-02)
  ==> 4.8332E+00 * exp[j(1.7957E+02)]
V5: -4.832E+00 +j( -4.089E-02)
  ==> 4.8325E+00 * exp[j(1.8048E+02)]
```

```
Solución en F=   1.0000000000E+05 Hz.
V1: 9.9663E-01 +j( -1.847E-03)
  ==> 9.9663E-01 * exp[j(-1.062E-01)]
V2: 9.9657E-01 +j( -1.739E-03)
  ==> 9.9657E-01 * exp[j(-1.000E-01)]
V3: 9.6926E-01 +j( -1.689E-03)
  ==> 9.6927E-01 * exp[j(-9.984E-02)]
V4: -4.833E+00 +j( 3.0792E-02)
  ==> 4.8326E+00 * exp[j(1.7963E+02)]
V5: -4.833E+00 +j( 2.3101E-02)
  ==> 4.8326E+00 * exp[j(1.7973E+02)]

Solución en F=   1.0000000000E+06 Hz.
V1: 9.9572E-01 +j( -1.842E-02)
  ==> 9.9589E-01 * exp[j(-1.060E+00)]
V2: 9.9566E-01 +j( -1.840E-02)
  ==> 9.9583E-01 * exp[j(-1.059E+00)]
V3: 9.6838E-01 +j( -1.787E-02)
  ==> 9.6854E-01 * exp[j(-1.057E+00)]
V4: -4.819E+00 +j( 2.7431E-01)
  ==> 4.8268E+00 * exp[j(1.7674E+02)]
V5: -4.819E+00 +j( 2.7354E-01)
  ==> 4.8268E+00 * exp[j(1.7675E+02)]

Solución en F=   1.0000000000E+07 Hz.
V1: 9.2285E-01 +j( -1.478E-01)
  ==> 9.3461E-01 * exp[j(-9.099E+00)]
V2: 9.2280E-01 +j( -1.478E-01)
  ==> 9.3456E-01 * exp[j(-9.098E+00)]
V3: 8.9759E-01 +j( -1.435E-01)
  ==> 9.0899E-01 * exp[j(-9.083E+00)]
V4: -3.735E+00 +j( 2.1985E+00)
  ==> 4.3339E+00 * exp[j(1.4952E+02)]
V5: -3.735E+00 +j( 2.1984E+00)
  ==> 4.3339E+00 * exp[j(1.4952E+02)]
```

La tensión de mayor interés es la del nodo 5, que es la salida del circuito. Si se estudia su valor modular (tercera columna de los resultados de salida), se puede ver como hasta pasados los 100 Hz no alcanza su valor estable (del orden de 4.8 V.). Esto, como es natural, es debido al acoplo en alterna, realizado a través de los condensadores C1 y C2.

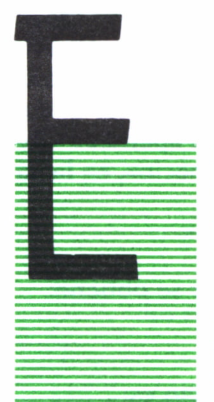
La ganancia se puede calcular para cada frecuencia, realizando la división V5/V1. Vemos cómo es aproximadamente del orden de 5 en las frecuencias centrales, con una fase de unos 180 grados (es un amplificador inversor), dato obtenido de la cuarta columna de los resultados.

Hasta muy alta frecuencia no se hacen notar los efectos de las capacidades internas del transistor Cbe y Cbc. Incluso a 10 MHz, máxima fre-

cuencia de nuestro análisis, la disminución de la salida es sólo de 0.5 V., mientras que en la fase ya es de 30 grados.

Como puede verse, el análisis en frecuencia es quizá una de las partes más sencillas del proceso de análisis de un circuito. No por ello es la menos importante. Normalmente su importancia es decisiva, sobre todo en caso de amplificadores y otros circuitos en los que se busca una respuesta de tipo lineal. El conocimiento a fondo de sus métodos de análisis es, por tanto, fundamental.

ANALISIS DE CIRCUITOS EN REGIMEN TRANSITORIO 5



N el análisis transitorio se busca calcular la evolución de la respuesta de un circuito, desde un instante de tiempo al siguiente, cuando a éste se le aplica una excitación repentinamente.

En caso de que el circuito sea lineal, el funcionamiento puede describirse por un conjunto de ecuaciones diferenciales, que, si los parámetros no varían con el tiempo, serán lineales y con coeficientes constantes.

Si los circuitos son no lineales, su descripción vendrá dada, en general, por un conjunto de ecuaciones diferenciales no lineales. En este caso los métodos de resolución analíticos son inútiles y se precisan técnicas numéricas.

Los métodos comúnmente empleados para atacar este problema se basan en el planteamiento de las *ecuaciones de estado* del circuito. Estas ecuaciones son las que relacionan entre sí las *variables de estado* de la red, que no son más que las corrientes en las bobinas y tensiones en los condensadores. Se denominan así porque sus derivadas con respecto del tiempo especifican cómo el «estado» en que se encuentra el circuito puede predecirse desde un instante de tiempo al siguiente.

Una vez que se tienen estas ecuaciones de estado se resuelve el sistema de ecuaciones diferenciales por un método numérico, tanto más complejo cuanto más alta sea la exactitud que necesitamos en las soluciones.

Otro procedimiento se basa en aplicar las fórmulas de integración numérica a las ecuaciones diferenciales que definen condensadores y resistencias, esto es, a sus relaciones corriente-tensión. La solución de esta integración numérica dará lugar a un circuito equivalente que podremos sustituir en la red, y proceder así al análisis en continua de un circuito muy simplificado.

Este es el método denominado del *circuito incremental*. Es el que se va a desarrollar en este capítulo, dejando los otros procedimientos para los

lectores de profundos conocimientos matemáticos que se sientan atraídos por la aventura del cálculo numérico. En la bibliografía citada al final de la obra encontrarán suficiente material para explorar en sus ratos de ocio.



ANALISIS TRANSITORIO DE CIRCUITOS LINEALES

A continuación vamos a realizar la integración de las ecuaciones corriente-tensión de los componentes básicos: condensadores, bobinas y resistencias.

Podemos emplear cualquier método de integración numérica. Cuanto mayor sea la precisión requerida, más complejo será el método a utilizar. En todo el desarrollo que sigue se emplea el método del trapecio, que el lector profano puede encontrar explicado con detalle en el apéndice «Resolución de una ecuación diferencial por el método del trapecio», en la página 131.

1. *Condensador*

Despejando la derivada en la ecuación del condensador se obtiene:

$$\frac{dv}{dt} = \frac{i}{C}$$

La solución de dicha ecuación por el método del trapecio es:

$$v_{n+1} = v_n + (\frac{1}{2C} i_{n+1} + \frac{1}{2C} 1_n) h =$$

$$= (v_n + F\frac{h}{\ } i_n) + \frac{h}{\ } i_{n+1}$$

$$v_{n+1} = Veq_n + Req_{n+1} \cdot i_{n+1} \text{,,} \qquad \begin{cases} Veq_n = v_n + \frac{h}{2C} i_n \\ Req_{n+1} = \frac{h}{2C} \end{cases}$$

Ecuaciones que dan como resultado el siguiente circuito equivalente incremental:

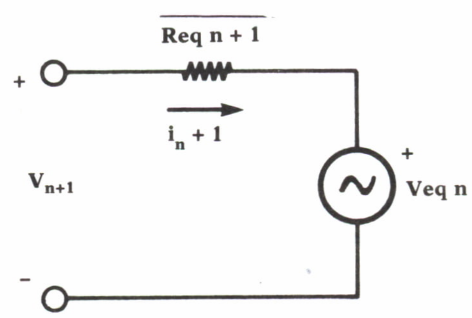


Fig. 43. Circuito incremental del condensador.

Ya que los valores de corriente y tensión en el instante *n* son constantes para el cálculo de los valores en el instante *n + 1*.

La resolución del problema requiere conocer los valores iniciales de tensión y corriente en el condensador para poder resolver el primer intervalo.

2. *Bobina*

Igual que con el condensador hacemos con la bobina:

$$\frac{di}{dt} = \frac{v}{L}$$

$$i_{n+1} = i_n + (\frac{1}{2L} v_{n+1} + \frac{1}{2L} v_n) h =$$

$$= (i_n + \frac{h}{2L} v_n) + \frac{h}{2L} v_{n+1}$$

$$i_{n+1} = Ieq_n + Geq_{n+1} \cdot v_{n+1} \text{ ,,} \qquad Ieq_n = i_n + \frac{h}{2L} v_n \qquad Geq_{n+1} = \frac{h}{2L}$$

Lo que nos da un circuito incremental equivalente de la siguiente forma:

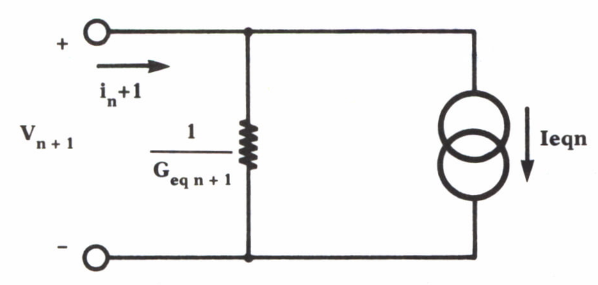


Fig. 44. Circuito incremental de la bobina.

Del mismo modo que para el condensador, se necesitan las condiciones iniciales de estado de la bobina para resolver el problema.

3. *Resistencia*

El circuito incremental de una resistencia es ella misma, ya que en su ecuación no se hallan involucradas derivadas de ningún tipo.

La sustitución de los componentes por sus circuitos equivalentes dará como resultado una red que sólo poseerá resistencias y generadores, de fácil análisis por el método nodal. Los coeficientes se determinarán en fun-

ción de la aproximación anterior. Con esto se consigue un sistema de ecuaciones con coeficientes reales, cuya solución son los valores de las incógnitas del circuito en el siguiente instante de tiempo. Este proceso se repite cuantas veces sea necesario.

Hay que resaltar la importancia que tiene el fijar un paso de integración adecuado. Un paso demasiado grande nos llevará a errores en la solución, que harán que se aparte totalmente de la realidad. Un paso demasiado pequeño, por otro lado, nos puede llevar a una duración del cálculo excesiva e innecesaria.

Como ejemplo se puede ver en la figura 45 un circuito lineal y su circuito incremental correspondiente, cuya obtención puede el lector justificar fácilmente.

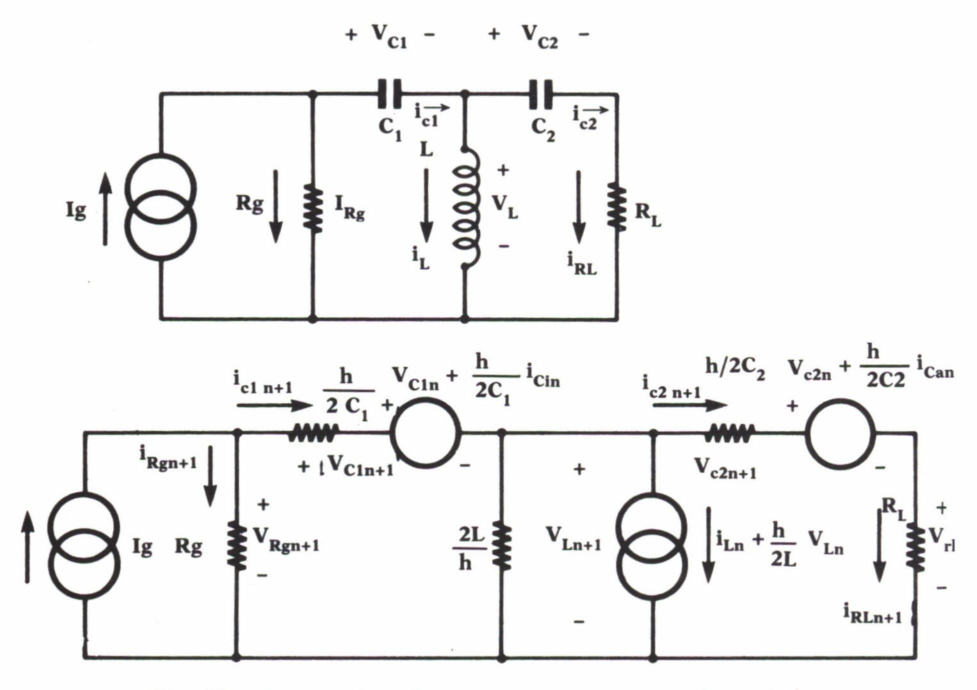


Fig. 45. Circuito lineal y su circuito incremental asociado.

CIRCUITO LINEAL Y SU CIRCUITO INCREMENTAL ASOCIADO

Los elementos no lineales se tratan del mismo modo que los lineales de cara al cálculo de la red incremental equivalente. La diferencia será que producirán una resistencia no lineal en ella, y, por tanto, el procedi-

miento de análisis tendrá que ser el visto en el capítulo «Análisis de circuitos en continua», en la página 45.

Tomemos como ejemplo el caso del diodo. Prescindiendo de efectos secundarios, lo trataremos como una resistencia no lineal, en la cual la relación corriente-tensión tendrá la siguiente forma:

$$id_{n+1} = Is \cdot (e^{V_{n+1}/Vt} - 1)$$

En la figura 46 se puede ver como ejemplo un circuito rectificador en media onda, con su circuito incremental equivalente. Dicho circuito se resolverá en continua por el método iterativo, ya explicado con detalle en el apartado correspondiente.

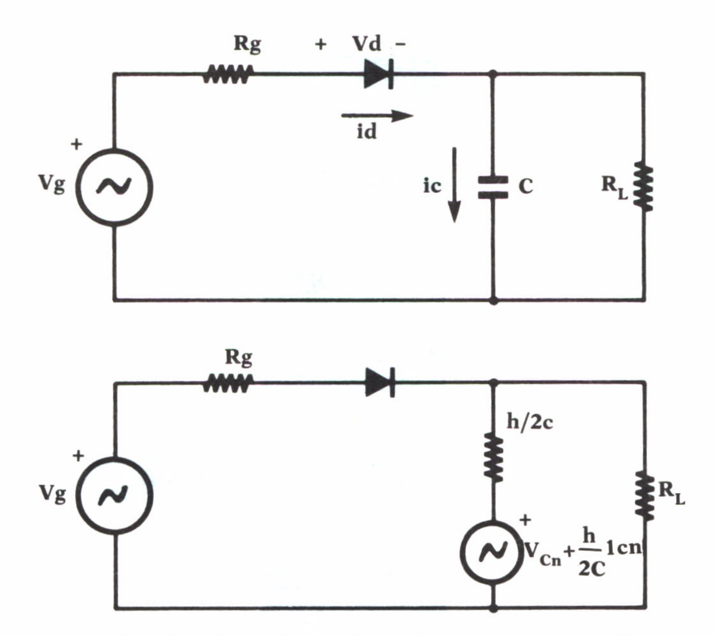


Fig. 46. Rectificador de media onda y circuito incremental asociado.

En el caso de que haya transistores en la red, lo primero que se debe hacer es sustituirlos por los circuitos equivalentes que nos ofrecen los modelos vistos para el análisis en continua, formados por diodos y generadores controlados, y proceder al análisis por el método general aquí establecido.

En este caso es preciso considerar las capacidades presentadas en dichos circuitos, que en el caso del análisis en continua se eliminaban. Como toda capacidad dará lugar a un circuito incremental equivalente como el de la figura 43.

El problema es que estas capacidades no son lineales, por lo que la resistencia a la que dan lugar en el circuito incremental tampoco lo será.

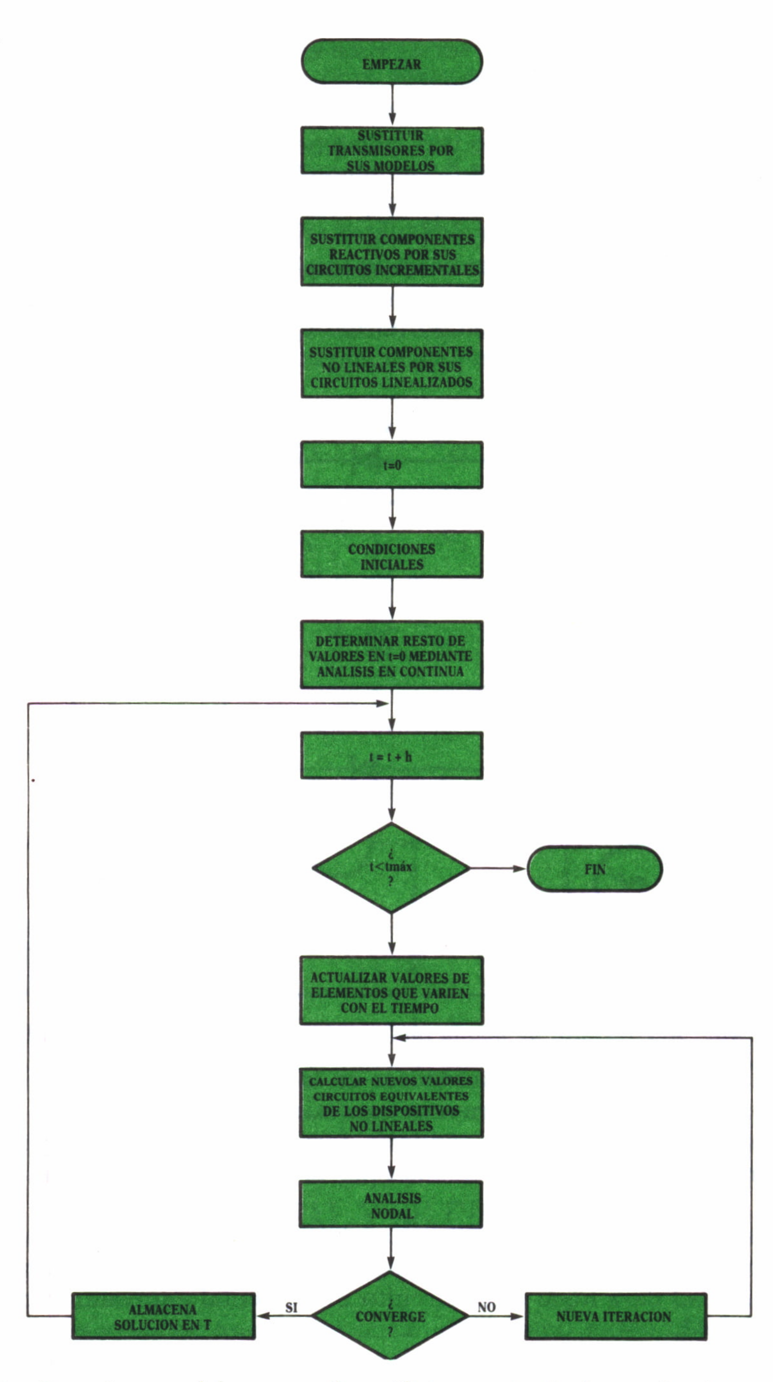


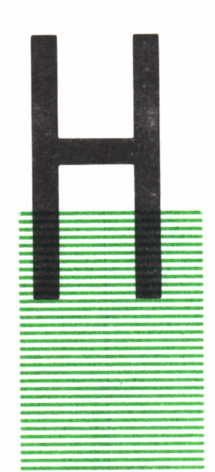
Fig. 47. Organigrama del proceso de análisis transitorio de un circuito no lineal.

El tratamiento del problema es en estos casos bastante más complicado. Normalmente, para una primera aproximación, los efectos capacitivos no se tienen en cuenta, usándose sólo cuando se quieren obtener unos resultados que se aproximen al máximo a la realidad.

El lector interesado puede encontrar en el apéndice «Estudio de la capacidad de una unión p-n», en la página 133, un estudio sobre la interpretación física de estas capacidades y el modo de tratarlas.

En la figura 47 puede verse, como colofón de este capítulo, un organigrama que resume el método de análisis transitorio de un circuito no lineal.

SENSIBILIDAD 6



HASTA ahora se han tratado los métodos básicos para analizar un circuito. El empleo del ordenador para realizar esta tarea resulta de gran utilidad, ya que permite abordar mayor complejidad y mayor volumen con el mismo esfuerzo.

Pero la solución a la que llegábamos no nos aportaba nada nuevo respecto a lo que suministraba el análisis tradicional. Gracias al ordenador podemos abarcar nuevos campos, como es el que se presenta en este capítulo, y que denominamos genéricamente *sensibilidad.*

Buscamos con este tipo de análisis poder predecir el comportamiento del circuito ante cambios en uno o más componentes del mismo. Este estudio persigue una doble finalidad:

• Prevención de cambios indeseados. Un circuito, durante el transcurso de su vida, puede ver variadas sus constantes por envejecimiento, efectos de radiación, cambios de temperatura, etc. Estas variaciones pueden hacer que se salga fuera de especificaciones, lo cual es una circunstancia indeseable en cualquier caso.

Por otro lado, un circuito que se va a fabricar en serie debe estar diseñado considerando que no todos los componentes que se monten en cada unidad van a ser iguales. Hay unas tolerancias de fabricación contra las que debemos estar prevenidos, para evitar que un elevado tanto por ciento de la producción resulte inútil.

• Introducción de cambios deseados. Partiendo de un circuito base nos podemos preguntar cómo lograr unas mejores especificaciones en sus características. Conociendo la sensibilidad de los parámetros a mejorar respecto a las diversas variables que componen el circuito, podremos llegar a deducir cuáles deben ser éstos para lograr el adecuado cambio en aqué-

llos. El estudio sistemático de estos métodos es lo que persiguen las técnicas de optimización.



DEFINICIONES

Consideremos como salida de un circuito el parámetro (tensión o corriente) cuyo estudio nos interesa.

Dicha salida se puede expresar como una función de los valores de sus parámetros internos (resistencias, bobinas, condensadores...). Llamaremos a estos parámetros internos de un modo genérico con la letra «σ», según lo cual:

$$F(\sigma 1, \sigma 2, \ldots \sigma n)$$

Su *sensibilidad* respecto a un determinado parámetro se define como el coeficiente que relaciona sus variaciones con la variación del mencionado parámetro.

Llamando F a la variable salida, y σi al parámetro que varía, podemos escribir:

$$\Delta F = S^F_{\sigma i} \,.\, \Delta\sigma i$$

$$S^F_{\sigma i} = \frac{\delta F}{\delta\sigma i}$$

Matemáticamente es, por tanto, la derivada parcial de la salida respecto al parámetro en cuestión.

A veces se suele utilizar en lugar de ésta la *sensibilidad relativa*, que relaciona variaciones porcentuales en vez de variaciones absolutas. La representaremos mediante una «s» minúscula. Se relaciona con la sensibilidad absoluta a través de la expresión:

$$s^F_{\sigma i} = \frac{\Delta F/F}{\Delta\sigma i/\sigma i} = \frac{\sigma i}{F} \,.\, \frac{\delta F}{\delta\sigma i} = \frac{\sigma i}{F} \,.\, S^F_{\sigma i}$$

En el caso particular de que la función F sea una función de la red en el dominio de la frecuencia, podemos ponerla en forma módulo-argumento de la siguiente forma:

$$F(j\omega) = |F(j\omega)| \,.\, e^{j\Phi}$$

En donde Φ será el argumento de F(jω). La sensibilidad absoluta la podremos expresar como:

$$S^F_{\sigma i} = \frac{\delta}{\delta\sigma i}\,(|F| \,.\, e^{j\Phi}) = \frac{\delta|F|}{\delta\sigma i} \,.\, e^{j\Phi} + |F|\, j\, e^{j\Phi}\, \frac{\delta\Phi}{\delta\sigma i}$$

$$S^F_{\sigma i} = e^{j\Phi} \,.\, S^{|F|}_{\sigma i} + j\,|F|\, e^{j\Phi} \,.\, S^{\Phi}_{\sigma i}$$

Y la sensibilidad relativa:

$$s_{\sigma i}^{F} = \frac{\sigma i}{F} \cdot S_{\sigma i}^{F} = s_{\sigma i}^{|F|} + j\,\Phi\, s_{\sigma i}^{\Phi}$$

Llamaremos *red adjunta o transpuesta* de una red dada a aquella cuya matriz de admitancias es la transpuesta de dicha matriz en la red original, es decir, la misma en la que se han sustituido filas por columnas.

Denominaremos a los parámetros de la red transpuesta con los mismos nombres que los de la red primitiva añadiéndoles «'». Según esto:

$$[Y'] = [Y]^{T} \Rightarrow Y'ij = Yji$$

Daremos sin demostración la siguiente metodología de cálculo de las sensibilidades:

Supongamos que tenemos un circuito con valor de salida Vo y queremos obtener su sensibilidad respecto a las variaciones de una admitancia Ya.

1. Se analiza el circuito de interés, obteniendo el valor de tensión y corriente en bornas de la admitancia Ya, a los que llamaremos Va e Ia.
2. Se analiza la red adjunta, con las siguientes modificaciones:

• Sustitución de los generadores independientes de corriente por circuitos abiertos.

• Sustitución de los generadores independientes de tensión por cortocircuitos.

• Introducción de un generador de corriente de valor unidad en el nodo de salida.

Con esto, el sistema a resolver tendrá como matriz de admitancias la transpuesta del sistema original, y como vector de generadores equivalentes

$$[Ieq] \;,,\; Ieqi = \begin{matrix} 0,\ \text{si el nudo i no es el de salida} \\ 1,\ \text{si el nudo i es el de salida.} \end{matrix}$$

Nos interesa calcular en esta red también corriente y tensión en la admitancia Ya. A estos valores les llamaremos V'a e I'a.

3. La sensibilidad se calculará como:

$$S_{Ya}^{Vo} = -Va \,.\, V'a$$

En caso de que deseásemos calcular la sensibilidad de Vo respecto a una impedancia Za, el proceso sería el mismo, pero la fórmula final para la sensibilidad sería:

$$S_{Ya}^{Vo} = -Ia \,.\, I'a$$

En principio vemos cómo el problema del cálculo de sensibilidades se ha reducido a la resolución de dos redes. Esta doble resolución, «a prio-

ri», requeriría resolver dos sistemas de ecuaciones y, por tanto, el doble de trabajo de ordenador.

La ventaja de este método se basa en que no es necesario realizar todo el proceso de solución del sistema de ecuaciones que genera la red adjunta. Si empleamos el método de descomposición L-U (ver «Resolución de sistemas de ecuaciones», en la página 133) tenemos:

$$[Y] . [V] = [I],$$

que resolvemos como:

$$[L] . [U] . [V] = [I]$$

Ahora bien: recordemos que [Y'] es la matriz transpuesta de la [Y]. Por tanto:

$$[Y'] = [Y]^T = ([L] . [U])^T = [U]^T . [L]^T$$

El sistema que tenemos que resolver será:

$$[Y'] [V'] = [I'],$$

que se resolverá como:

$$[U]^T . [L]^T . [V'] = [I']$$

El proceso se ha simplificado, ya que no es preciso descomponer la matriz de la red adjunta teniendo descompuesta la de la red principal. Será necesario realizar tan sólo la marcha inversa:

$$[U]^T . [y] = [I'];$$

$$[L]^T . [V'] = [y]$$



EJEMPLO DEL METODO DE CALCULO DE SENSIBILIDADES

Para ilustrar el método de cálculo se va a resolver un sencillo ejemplo. Tenemos el circuito de la figura:

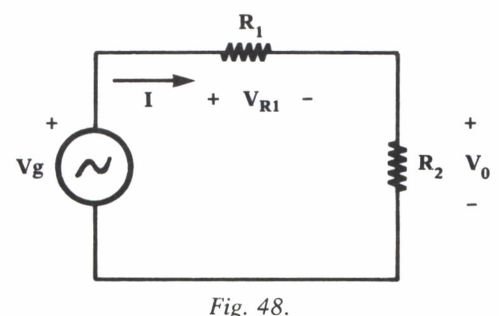


Fig. 48.

Deseamos conocer la sensibilidad de la tensión de salida Vo en función de las variaciones de la resistencia R1. Resolviendo directamente el sencillo problema tendremos:

$$Vo = Vg \frac{R2}{R1 + R2} \quad ,, \quad Vr1 = Vg \frac{R1}{R1 + R2} \qquad Ir1 = \frac{Vg}{R1 + R2}$$

La sensibilidad la podemos calcular directamente a partir de su definición en este sencillo caso:

$$S_{R1}^{Vo} = \frac{\delta Vo}{\delta R1} = - \frac{Vg\, R2}{(R1 + R2)^2}$$

Probemos ahora con la red adjunta o transpuesta. El circuito a resolver, según las reglas establecidas, será el siguiente:

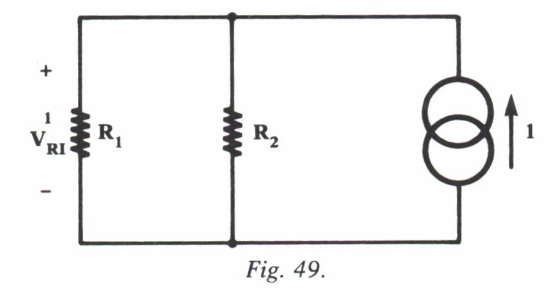


Fig. 49.

Nos interesa conocer de este circuito la corriente que atraviesa la resistencia R1:

$$I'r1 = \frac{1 \, . \, R2}{R1 + R2}$$

Y a continuación podemos obtener la sensibilidad respecto a R1 de la forma:

$$S_{R1}^{Vo} = -Ir1 \, . \, I'r1 = - \frac{Vg}{R1 + R2} \; \frac{R2}{R1 + R2} = - \frac{Vg\, R2}{(R1 + R2)^2}$$

Que coincide con la que habíamos encontrado por el método directo.

Basándose en los principios establecidos en este capítulo, la tarea de realizar un programa para cálculo de sensibilidades utilizando los programas hasta ahora presentados no debe presentar ningún problema al lector. Por tanto, se deja a su imaginación esta tarea, que puede servir como ejercicio para evaluar si realmente los conocimientos presentados se han adquirido de forma razonada y duradera.

OPTIMIZACION DE CIRCUITOS 7

UNA vez que tenemos las herramientas básicas de análisis de cirucitos, vamos a abordar de forma general el problema de la síntesis. El organigrama general de este proceso ya se presentó en la figura 2. En este capítulo nos vamos a centrar en el proceso iterativo que se debe llevar a cabo una vez definidos unos valores de componentes y analizado el circuito. Son varias las cuestiones que debemos responder llegados a ese punto del diseño:

- ¿Qué criterio se debe seguir para determinar si el circuito obtenido en el último análisis satisface nuestras exigencias o no?
- ¿Qué componentes se deben alterar para acercarnos a las especificaciones buscadas?
- ¿Cuál debe ser la magnitud y el sentido de ese cambio?

En el diseño tradicional, estas preguntas se respondían en base a la experiencia del diseñador, que «intuía» el cambio necesario para hacer funcionar el equipo de acuerdo a lo establecido «a priori». La complejidad de cualquier método de cálculo hacía imposible abordar el problema de forma sistemática.

El ordenador soluciona este problema, y en torno a él han surgido métodos para lograr realizar estos pasos de forma automática. El conjunto de estos métodos es lo que se conoce como *técnicas de optimización* de circuitos.

El diseño de acuerdo a estas técnicas tiene tres pasos fundamentales:

1. Definir una función de comportamiento que sea una medida general del error entre la respuesta actual y la deseada de la red.
2. Definir una red inicial, que será el punto de partida para el proceso de diseño.
3. Analizar la red y ajustar los parámetros de diseño deseados y la estructura de forma que se minimice la función de error. Este proceso de-

berá repetirse hasta que la función de error se haya reducido por debajo de un cierto nivel o bien hasta que no se pueda reducir más.

En este proceso vemos la aparición de dos nuevos problemas que aún no hemos tratado:

- Cómo determinar la función de comportamiento o de error de la red.
- Qué métodos emplear para encontrar sus valores mínimos.

Estos dos temas son los que vamos a tratar en el resto del capítulo.



DETERMINACION DE LA FUNCION DE COMPORTAMIENTO O DE ERROR

La determinación apropiada de la función de error es fundamental en todo problema de optimización de circuitos. Si está mal definida, o no tiene en cuenta todas las variables posibles, el resto del procedimiento nos conducirá a unos resultados carentes de sentido físico.

Con el fin de simplificar la notación a emplear llamaremos «σ» a un vector que contendrá los parámetros del diseño, es decir, los valores actuales de los componentes. Será un vector en columna, a efectos del cálculo matricial, esto es:

$$\sigma = \begin{matrix} \sigma 1 \\ \sigma 2 \\ \dots \\ \sigma n \end{matrix} \quad ; \quad \sigma^T = (\sigma 1, \sigma 2, \dots, \sigma n)$$

La función de error más usada en los problemas de optimización es la de *mínimos cuadrados* Puede expresarse de forma general como:

$$E(\sigma) = \int_{\{H\}} |W(\varphi) \, . \, e(\sigma,\varphi)|^2 \, d\varphi \qquad (1)$$

Expliquemos detalladamente cada miembro de esta igualdad:

- φ es una variable sobre la cual se optimiza el circuito. Puede ser la frecuencia, el tiempo, la temperatura...
- e(σ,φ) es el error entre la respuesta actual y la deseada de la red. Como se ve, es función del vector de componentes y de la variable sobre la que actuamos.
- W(φ) es una función de peso. Con ella queremos representar que no todos los valores de la variable φ nos interesan de igual manera a la hora de calcular el error. Así, habrá unas zonas de mayor importancia, que llevarán un valor de la función de peso mayor, y otras de importancia menor, en las que W(φ) será de menor valor.

• {H} es el campo de variación de la variable φ, es decir, el rango en el cual nos interesa optimizar la función.

• La integral a lo largo de todo el campo nos indica que vamos a sumar todos los errores en esa zona, y que ese total es lo que buscamos que sea mínimo. Se emplea una función cuadrática para evitar que al realizar esta integración los errores negativos se compensen con los positivos, dando como resultado un error nulo falso.

Para aclarar mejor el sentido de cada uno de los términos de esta función de error vamos a estudiar un ejemplo de forma gráfica. Se trata de un filtro paso alto.

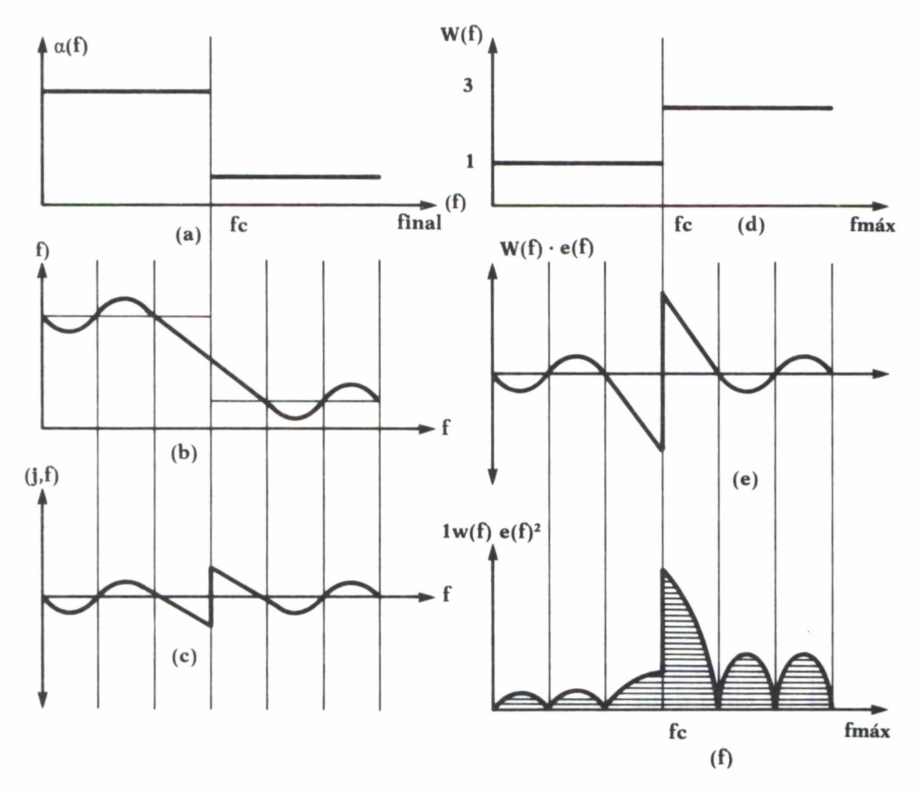


Fig. 50. Función de error para un filtro paso alto.

La figura (a) presenta las especificaciones de diseño. Se desea tener una atenuación de 100 en la banda eliminada y de 10 en la banda de paso. Estas especificaciones se deben cumplir hasta una frecuencia fmáx.

De lo dicho hasta ahora ya podemos decir, por pura analogía, cuáles son en este caso dos de los elementos componentes de la función de error:

- La variable φ va a ser la frecuencia, que representaremos como *f*.
- El rango {H} es el intervalo de frecuencias de 0 a fmáx.

En la figura (b) se ha superpuesto a la función que representaba las especificaciones de diseño la función obtenida en la primera aproximación. Como vemos, no coinciden, y en la figura (c) se representa la diferencia entre ellas, que constituye el error, designado en (1) por e(σ,φ).

Continuemos analizando el ejemplo. El error que más nos interesa minimizar en este caso es el cometido en la banda pasante, ya que la señal que se vea afectada por él es señal útil de la salida del circuito. Introducimos para ello una función de peso, W(f), representada en la figura (d), mediante la cual triplicamos el error para f > fc. En (e) se ve el efecto multiplicador señalado.

Por último, al elevar al cuadrado convertimos todos los errores en positivos, y obtenemos como resultado de la integral el total del área rayada en (f).

Normalmente el rango de valores permitidos de la variable independiente es discreto en lugar de continuo. En este caso la integral se convierte en una suma, y tendremos:

$$E(\sigma) = \sum_{i \in \{H\}} |W_i \cdot e(\sigma)_i|^2 \qquad (1)$$

A esta función se le denomina *función objetivo*, ya que su mínimo nos dará la mejor solución posible. *i* representa cada uno de los puntos de la variable φ que debemos considerar para la optimización.



METODOS PARA HALLAR EL MINIMO DE LA FUNCION OBJETIVA

Casi todos los métodos numéricos para determinar el óptimo de una determinada función son iterativos. Parten de una solución inicial dada y, a partir de ella, van generando una serie de estimaciones de dicho vector. Cada una de ellas debe presentar una mejora respecto a la anterior en lo que se refiere al valor de la función objetivo que proporciona.

Los distintos métodos difieren entre sí en la estrategia para producir la serie de estimaciones. Podemos establecer dos tipos fundamentales de métodos:

- Métodos de búsqueda directa: Se basan únicamente en comparación de valores de la función objetivo. Suelen ser simples y fáciles de programar, pero lentos.

• Métodos de gradiente: Usan las derivadas de la función objetivo para lograr una convergencia más rápida. Los hay de primero y segundo orden, según que utilicen sólo las primeras derivadas o las primeras y las segundas, respectivamente.



METODOS DE BUSQUEDA DIRECTA

En ellos se trata de reducir el valor de la función objetivo E usando comparaciones del valor que toma en una estimación con los valores en puntos próximos. Esto es:

• Dado un componente del circuito, se estudia en qué direccion debe variar su valor (aumentar o disminuir) para producir un descenso en la función objetivo.

• Este proceso se repite para cada uno de los componentes, obteniendo así una «dirección global de búsqueda».

• Una vez obtenida esta dirección se realiza un paso en ella, esto es, se actualizan los valores de los componentes en la dirección elegida y se calcula la nueva función objetivo.

• Estos pasos se repiten hasta que las variaciones de la función objetivo que se logren en cada paso sean menores que un valor máximo de error fijado con antelación.

Hay dos formas básicas de plasmar este procedimiento:

• Mediante una minimización independiente para cada parámetro.

- Se comienza con una variable y se prosigue con ella hasta que sus variaciones en ambos sentidos aumenten la función de error.
- Este proceso se repite para el resto de las variables.
- Una vez realizado con todas se vuelve a repetir el ciclo completo hasta que al terminar uno de ellos no se haya conseguido una mejora por encima del margen de error.

Este ajuste puede ser, por tanto, muy complejo si existen interacciones entre los distintos elementos de cara a la función objetivo.

• Por el procedimiento de variables alternadas. En lugar de optimizar al máximo para cada variable, se realiza un paso en una de ellas y se pasa a evaluar la dirección a continuar en la siguiente. Este proceso se repite cíclicamente hasta llegar a un valor de la función de error imposible de minimizar.

Hay otros muchos métodos de optimización por búsqueda directa, para cuyo estudio se remite al lector a la bibliografía especializada.

METODOS DE GRADIENTE

Si bien estos métodos conducen a unos resultados aceptables de una forma más rápida, son mucho más difíciles en su concepción, por lo que aquí se van a tratar de forma muy elemental.

Estos métodos se basan en el desarrollo de la función de error del circuito, que hemos denominado e(σ), en serie de Taylor en un entorno de la solución actual. Con eso podremos obtener:

$$e(\sigma + \Delta\sigma) = e(\sigma) + g^T . \Delta\sigma + \frac{1}{2} \Delta\sigma^T . [H] . \sigma$$

en donde:

$$g^T = (\frac{\delta e}{\delta\sigma 1} + \frac{\delta e}{\delta\sigma 2} + ... + \frac{\delta e}{\delta\sigma 1})$$

$$[H] = \begin{matrix} \frac{\delta^2 e}{\delta\sigma 1^2} & \frac{\delta^2 e}{\delta\sigma 1 \delta\sigma 2} & ... & \frac{\delta^2 e}{\delta\sigma 1 \delta\sigma n} \\ \frac{\delta^2 e}{\delta\sigma 2 \delta\sigma 1} & \frac{\delta^2 e}{\delta\sigma 2^2} & ... & \frac{\delta^2 e}{\delta\sigma 2 \delta\sigma n} \\ ... & ... & ... & ... \\ \frac{\delta^2 e}{\delta\sigma n \delta\sigma 1} & \frac{\delta^2 e}{\delta\sigma n \delta\sigma ds2} & ... & \frac{\delta^2 e}{\delta\sigma n^2} \end{matrix}$$

g es el *gradiente o jacobiano* de la función y [H] el *Hessiano*. Con este desarrollo tenemos:

• Métodos de primer orden: Toman sólo los dos primeros términos, sin considerar el que contiene la matriz [H]. El más conocido es el llamado método del descenso más rápido.

En este método, a partir de una solución dada σi, se obtiene la siguiente restando a ésta el producto de una constante por el valor del vector gradiente en el punto σi:

$$\sigma_{i+1} = \sigma_i - K_i . g(\sigma_i)$$

La constante puede variar de una iteración a otra, y se determinará de forma que σ + 1 dé como resultado una función objetivo menor que σi.

El cálculo de las derivadas que componen la matriz jacobiana lo podemos realizar de muy diferentes maneras. Para ilustrar la más cómoda para nosotros y que nos permite aplicar conocimientos que ya hemos adquiri-

do, reconsideremos el significado que la función e(σ) tiene en nuestro circuito. Es la diferencia entre nuestra salida y la salida que deseamos, esto es:

$$e(\sigma) = V(\sigma) - Vop$$

En donde V puede ser cualquier función de la red (tensión o corriente) en un nodo o en un determinado componente. Vop es constante respecto a los parámetros de diseño. Por tanto, al calcular la derivada parcial tendremos:

$$\frac{\delta e}{\delta \sigma i} = \frac{\delta}{\delta \sigma i} [V(\sigma) - Vop] = \frac{\delta V(\sigma)}{\delta \sigma i} = S^{V}_{\sigma i}$$

Expresión de la sensibilidad de la función de salida respecto al parámetro de la red σi.

Para calcular el jacobiano de la función bastará por tanto hallar las sensibilidades de la red respecto a todos los componentes que la forman, que se pueden obtener fácilmente por el método de la red adjunta.

• Métodos de segundo orden: toman el desarrollo en serie de Taylor incluyendo el Hessiano. De los múltiples algoritmos existentes vamos a exponer la metodología de cálculo de uno de los más conocidos y más eficientes: el de Davidon-Fletcher-Powell.

Este método no utiliza puramente la matriz Hessiana, sino que emplea una matriz que llamaremos ¿Sñ, que se actualiza en cada iteración. El proceso para la iteración [I] es el siguiente:

1. Se calcula la dirección de búsqueda haciendo

$$d_i = -[S_i] \, . \, g(\sigma_i)$$

En la primera iteración se toma para [S] la matriz identidad, con lo que la dirección de búsqueda coincide con la de los métodos de primer grado.

2. Se efectúa una búsqueda del mínimo de la función objetivo en esa dirección. Esto se dará para un valor:

$$\sigma_{i+1} = \sigma_i + K_i \, . \, d_i = \sigma_i + p_i$$

En otras palabras, buscamos un valor de K tal que E(σi + 1) sea el mínimo en la dirección di. Este mínimo se calcula por búsqueda directa, dando valores a K y calculando el valor de la función objetivo.

3. Calcular el nuevo jacobiano de la función en este nuevo punto, g(σi + 1).

4. Obtener la diferencia entre los jacobianos de las dos iteraciones:

$$q_i = g(\sigma_{i+1}) - g(\sigma_i)$$

5. Calcular la nueva matriz ¿Sñ mediante la fórmula:

$$[S_{i+1}] = [S_i] + [A_i] + [B_i],$$

en donde

$$[A_i] = \frac{p_i \cdot p_i^T}{p_i^T \cdot q_i} \quad y \quad [B_i] = \frac{[S_i] \cdot q_i \cdot q_i^T \cdot [S_i]}{q_i^T \cdot [S_i] \cdot q_i}$$

La complejidad de estas fórmulas está más en la forma de escribirlas que en el cálculo necesario para obtenerlas. Notar, por ejemplo, que los denominadores que aparecen en las expresiones de las matrices [Ai] y [Bi] son números, no matrices, con lo cual no hay que realizar ninguna inversión. Por ejemplo, en la matriz [Ai]:

$$p_i^T \cdot q_i = p_1 \cdot q_1 + p_2 \cdot q_2 + \ldots + p_n \cdot q_n$$

6. Este procedimiento se repetirá desde el paso (1) hasta que cada componente de di sea menor que una cantidad prefijada.

El hecho de que este método sea de segundo orden es porque la serie de matrices [Sk] que calculamos tiene por límite el Hessiano de la función. La interpretación de dicha serie es la siguiente:

• Para las primeras iteraciones, ¿Sñ es la matriz identidad, con lo cual la búsqueda se hace por el método del descenso más rápido. Este método es bueno cuando se está relativamente lejos de la solución, pero es lento en proximidades de dicho punto. Por eso:
• En las iteraciones finales, [S] se aproxima al valor del Hessiano de la función. Estamos empleando un método de segundo orden, muy bueno en las proximidades de la solución, pero con problemas para valores lejanos. Este sistema aprovecha lo mejor de los métodos de primer y segundo orden.



CONCLUSIONES

Este capítulo no es sino un brevísimo repaso de las técnicas de optimización de circuitos para dar una idea de su complejidad y de los problemas que plantean. Es cierto que están más cerca que ningún otro sistema de la automatización total del diseño, pero necesitan una interacción muy fuerte con el usuario, que es quien debe determinar:

• Estructura de la red y primera solución.
• La función objetivo más adecuada para resolver el problema.
• El algoritmo de búsqueda del mínimo utilizado.

Una vez establecido esto no ha terminado el trabajo, pues muchas veces los métodos empleados presentan difíciles problemas de convergencia, estabilidad, etc., que pueden hacer necesaria su sustitución a partir de una determinada iteración.

APENDICE A. ESTUDIOS ESPECIALES



ANALISIS NODAL DE UN CIRCUITO

Se considera, para simplificar el desarrollo, que el circuito a analizar esté compuesto de los siguientes elementos:

- Resistencias.
- Bobinas.
- Condensadores.
- Fuentes de corriente independientes.
- Fuentes de corriente controladas por tensión.

El circuito consta de *m* ramas y *n* nudos.

Una rama generalizada tendrá la siguiente estructura:

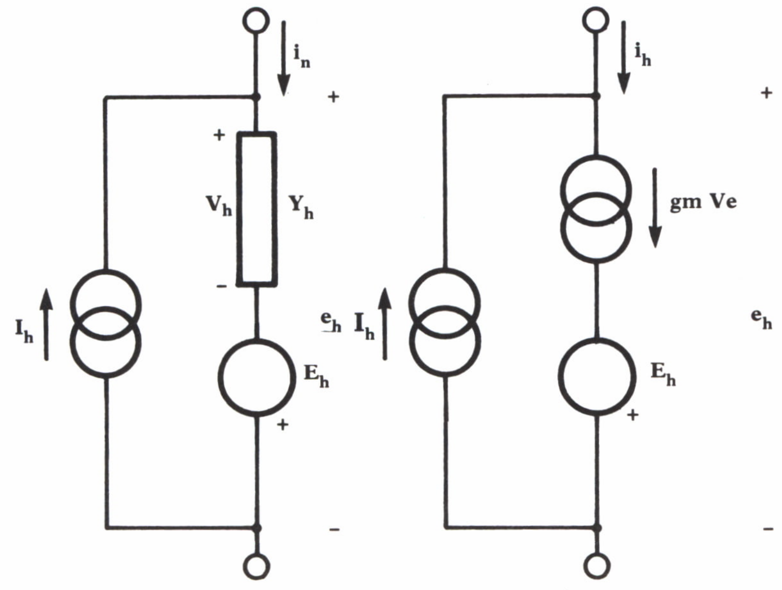


Fig. 51. Rama generalizada. Convenio de signos.

La rama generalizada consta, por tanto, de

- Un generador independiente de corriente.
- Un generador independiente de tensión.
- Una admitancia pasiva (o un generador de corriente dependiente).

Las ecuaciones básicas de esta rama, considerada como la número *h* del circuito, serán:

$$eh = Vh - Eh$$
$$ih = -Ih + jh = -Ih + Yh\ (eb + Eb) + gm\ (el + El) \quad (1)$$

A continuación se definen las siguientes matrices:

[Igen]: Generadores de corriente. Dimensión: m × 1. Cada Igen*i* representa el valor del generador de corriente asociado a la rama *i*.
[Egen]: Generadores de tensión. Igual dimensión y significado que ¿Igenñ.
[e]: Vector de tensiones en cada rama. Dimensión: m × 1.
[i]: Vector de corrientes en cada rama. Dimensión: m × 1.
[Y] Matriz de admitancias de rama. Dimensión: m × m. Un elemento de esta matriz Yij tendrá por valor

- Yi, si i = j (Admitancia de la rama i).
- 0, si i <> j.

[gm]: Matriz de conductancias de los generadores de corriente controlados por tensiones de rama. Dimensión: m × m. Un elemento genérico gmij tendrá por valor:

- gmi, si el generador de corriente se halla en la rama i y está controlado por la tensión en la rama j.
- 0, en caso contrario.

[Vn]: Vector de tensiones de nudo. Dimensión n × 1.
[A]: Matriz de incidencia. Representa la geometría del circuito. Dimensión: n × m. Sus elementos Aij tienen como valor:

- 0, si la rama j no está conectada al nudo i.
- 1, si la corriente de la rama j sale del nudo i.
- −1, si la corriente de la rama j entra al nudo i.

De las definiciones de las matrices se pueden establecer las siguientes relaciones:

$$[e] = [A]^T . [Vn] \quad (2)$$
$$[A] . [i] = 0, \quad (3)$$

según la ley de Kirchoff de corrientes.

Por otra parte, el sistema de ecuaciones de corrientes de (1) de todas las ramas del circuito se puede expresar en forma matricial como:

$$[i] = -[Igen] + [Y] . [e] + [Y] . [Egen] + [gm] . [e] + [gm] . [Egen] \quad (4)$$

considerando la existencia de admitancias y generadores controlados simultáneamente. Sustituyendo (4) en (3) se obtiene:

$$-[A] . [Igen] + [A] . [Y] . [e] + [A] . [Y] . [Egen] + [A] . [gm] . [e] + \\ + [A] . [gm] . [Egen] = 0$$

Transformando esta ecuación:

$$[A] . [Y+gm] . [e] = -[A] . [Y+gm] . [Egen] + [A] . [Igen]$$

Sustituyendo (2) en esta última ecuación obtenemos:

$$[A] . [Y+gm] . [A]^T . [Vn] = -[A] . [Y+gm] . [Egen] + [A] . [Igen]$$

Ecuación que se puede presentar de la forma:

$$[Yn] . [Vn] = [Ieq] \qquad (5)$$

En donde

$$[Yn] = [A] . [Y + gm] . [A]^T$$

es la matriz de admitancia nodal reducida (dimensión n × n)

$$[Ieq] = -[A] . [Y+gm] . [Egen] + [A] . [Igen]$$

es el vector de generadores de corriente equivalentes (dimensión n × 1)

Las ecuaciones representadas en notación matricial en (5) son las ecuaciones nodales de un circuito.

Para aclarar el procedimiento a seguir se va a realizar un ejemplo. El circuito de la figura es el equivalente en pequeña señal de un transistor conectado en emisor común:

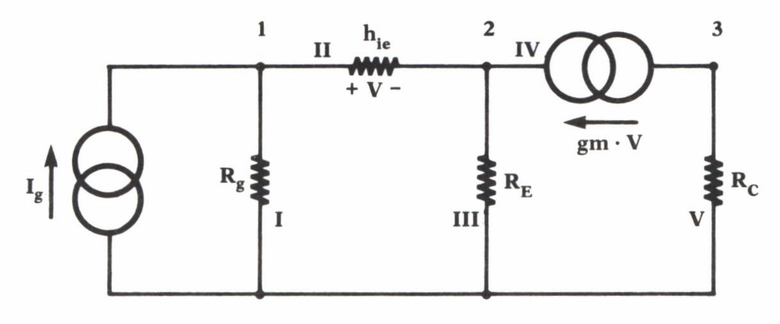


Fig. 52. Ejemplo de cálculo.

El circuito consta de cinco ramas y tres nudos. En la figura, las ramas se han indicado con numeración romana. Las matrices de interés son las siguientes:

• Matriz de generadores equivalentes: Sólo posee un término, ya que sólo hay un generador independiente:

$$[Ieq] = \begin{matrix} Ig \\ 0 \\ 0 \end{matrix}$$

• Matriz de incidencia:

$$[A] = \begin{matrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{matrix}$$

• Matriz de admitancias:

$$[Y] = \begin{matrix} 1/Rg & 0 & 0 & 0 & 0 \\ 0 & 1/hie & 0 & 0 & 0 \\ 0 & 0 & 1/Re & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/Rc \end{matrix}$$

• Matriz de transconductancias:

$$[gm] = \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & gm & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Cálculo de la matriz de admitancia nodal reducida:

$$[Yn] = [A] . [Y + gm] . [A]^T =$$

$$= \begin{matrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{matrix} \quad \begin{matrix} 1/Rg & 0 & 0 & 0 & 0 \\ 0 & 1/hie & 0 & 0 & 0 \\ 0 & 0 & 1/Re & 0 & 0 \\ 0 & gm & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/Rc \end{matrix} \quad \begin{matrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{matrix} \quad =$$

$$= \begin{matrix} 1/Rg & 1/hie & 0 & 0 & 0 \\ 0 & -1/hie - gm & 1/Re & 0 & 0 \\ 0 & gm & 0 & 0 & 1/Rc \end{matrix} \quad \begin{matrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 1 \end{matrix}$$

$$= \begin{matrix} 1/Rg + 1/hie & -1/hie & 0 \\ -1/hie - gm & 1/hie + 1/Re + gm & 0 \\ gm & -gm & 1/Rc \end{matrix}$$

Esta matriz se puede descomponer en otras cuatro que muestran la aportación de cada una de las ramas:

$$[Yn] = \begin{matrix} 1/Rg & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} + \begin{matrix} 1/hie & -1/hie & 0 \\ -1/hie & 1/hie & 0 \\ 0 & 0 & 0 \end{matrix} +$$

$$+ \begin{matrix} 0 & 0 & 0 \\ -gm & gm & 0 \\ gm & -gm & 0 \end{matrix} + \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1/Rc \end{matrix}$$

Que responden a las reglas establecidas en el capítulo «Fundamentos del diseño de circuitos asistido por ordenador», en la página 13.

El sistema de ecuaciones obtenido es, por tanto:

$$\begin{matrix} 1/Rg + 1/hie & -1/hie & 0 \\ -1/hie - gm & 1/hie + 1/Re + gm & 0 \\ gm & -gm & 1/Rc \end{matrix} \quad \begin{matrix} V1 \\ V2 \\ V3 \end{matrix} = \begin{matrix} Ig \\ 0 \\ 0 \end{matrix}$$

O bien en la forma habitual:

$$(1/Rg + 1/hie)\ V1 - (1/hie)\ V2 = Ig$$
$$(-1/hie - gm)\ V1 + (1/hie + 1/Re + gm)\ V2 = 0$$
$$gm\ V1 - gm\ V2 + (1/Rc)\ V3 = 0$$



RESOLUCION DE UNA ECUACION DIFERENCIAL POR EL METODO DEL TRAPECIO

Se desea resolver la ecuación diferencial en forma explícita:

$$y' = f(x)$$

De la cual conocemos un punto, que llamaremos (X0, Y0). Este punto es la *condición inicial* del problema.

Buscamos obtener el valor de la función y para un determinado conjunto finito de valores de x (x1, x2, x3...) equidistantes entre sí, de tal manera que:

$$x2 - x1 = x3 - x2 = x4 - x3 = ... = h$$

h recibe el nombre de *paso de integración.*

Como sabemos, la derivada de una función nos representa la pendiente de la recta tangente a ella en un determinado punto. Nuestra ecuación diferencial nos da, por tanto, el valor de las tangentes a la función que buscamos en cada uno de sus puntos.

Todos los métodos de resolución numérica de ecuaciones diferenciales se basan en el mismo principio: sustituir la función que no conocemos por una recta cuya pendiente sea el promedio de las pendientes de las tangentes a la curva desconocida en distintos puntos internos al intervalo (xi, xi + h).

En el caso del método del trapecio, este promedio es simple de calcular, y es igual a la media aritmética de los valores de las derivadas en los puntos extremos del intervalo. Matemáticamente:

$$y_{i+1} = y_i + \frac{y'(x_i) + y'(x_{i+1})}{2} (x_{i+1} - x_i)$$

La interpretación geométrica de este método la podemos ver en la figura 53.

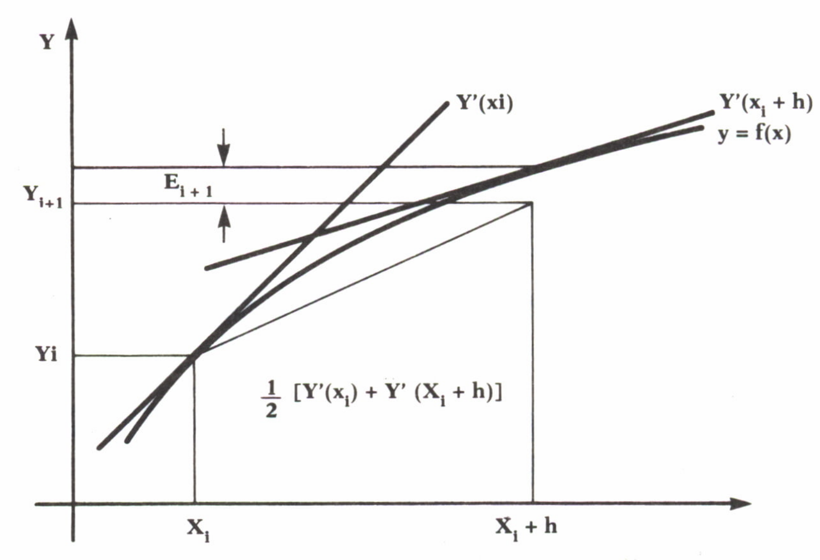


Fig. 53. Interpretación gráfica del método del trapecio.

El punto yi + 1 es la solución que nos da nuestro método. Si la curva real fuese la dibujada, tendríamos un error en esta aproximación de valor Ei + 1, que además se iría acumulando en el siguiente intervalo.

Como hemos fijado que el paso de integración sea constante, podemos poner la expresión del método de la forma:

$$y_{i+1} = y_i + \frac{y'(x_i) + y'(x_{i+1})}{2} h$$



ESTUDIO DE LA CAPACIDAD DE UNA UNION P-N

En todo dispositivo de unión p-n existen unas acumulaciones de portadores en su volumen, irregularmente distribuidos. Estas acumulaciones varían de forma no lineal con la tensión que se aplique al dispositivo.

El fenómeno se manifiesta al exterior como un efecto capacitivo. Puesto que la variación de carga no es lineal, la capacidad equivalente que presenta el dispositivo tampoco lo será.

Llamemos Cpn a esta capacidad. Se puede considerar constituida por otras dos básicas, fruto de dos fenómenos físicos distintos:

$$Ct = Cd + Cz$$

- Cd: Capacidad de difusión. Se debe a la existencia de portadores de corriente en el volumen del componente, lejos de la zona de la unión. Este es el efecto fundamental bajo polarización directa. La expresión para ella es:

$$Cd = \frac{\tau t}{Vt} Is \left(e^{v/Vt} - 1\right)$$

donde τt es el tiempo de tránsito de los portadores.

- Cz: Capacidad en la zona de carga espacial o zona de la unión pn. Su importancia relativa es mayor bajo polarización inversa. Se relaciona con la tensión en los extremos del diodo a través de la expresión:

$$C = \frac{K}{\sqrt{\Phi_{\cdot 0}} - V}$$

$\Phi_{\cdot 0}$ es la llamada tensión de banda plana del dispositivo y K una constante que depende de la geometría del componente.

El circuito incremental de una capacidad no lineal es exactamente el mismo que el del caso lineal, excepto que la resistencia de valor constante obtenida allí será no lineal:

$$Veq_n = v_n + \frac{h}{2C(Vn)} i_n$$

$$Req_{n+1} = \frac{h}{2C(Vn + 1)}$$

Para analizar este componente deberemos sustituirlo por una rama linealizada, según se estudió en el apartado «Análisis de circuitos en continua», en la página 45.



RESOLUCION DE SISTEMAS DE ECUACIONES

Muchos son los métodos para resolver sistemas de ecuaciones. En este capítulo sólo se va a tratar uno de ellos, debido a su uso generalizado en los programas de análisis de circuitos. Este método es el llamado de Choleski o de la descomposición L-U.

Su fundamento es muy sencillo. Tenemos el sistema:

$$[A] . [X] = [B]$$

Que es de orden *n*. Supongamos que podemos descomponer la matriz cuadrada [A], de orden $n \times n$ en el producto de otras dos matrices:

$$[A] = [L] . [U]$$

Si esto es posible, el sistema de ecuaciones que tenemos lo podemos representar por:

$$[L] . [U] . [X] = [B]$$

Basándonos en esto, podemos descomponer el sistema de ecuaciones en otros dos, de forma que:

$$[U] . [X] = [Y] \qquad (1)$$
$$[L] . [Y] = [B] \qquad (2)$$

Con esto, hemos descompuesto el problema en la resolución de dos sistemas. Primero habrá que resolver el (2) y, con el vector [T] que allí se obtiene, se resolverá el sistema (1).

¿Dónde está la ventaja? Hasta ahora lo único que se ha conseguido es duplicar aparentemente el problema.

La gran ventaja de esta descomposición es que se buscan las matrices [l] y [u] de forma que sean *triangulares*, esto es, de la forma:

$$[L] = \begin{array}{ccccc} l11 & 0 & 0 & \ldots & 0 \\ l21 & l22 & 0 & \ldots & 0 \\ l31 & l32 & l33 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \\ ln1 & ln2 & ln3 & \ldots & lnn \end{array}$$

$$[U] = \begin{array}{ccccc} 1 & u12 & u13 & \ldots & uln \\ 0 & 1 & 123 & \ldots & uln \\ 0 & 0 & 1 & \ldots & 13n \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \end{array}$$

Los sistemas de ecuaciones con matrices triangulares son de resolución sencilla. Basta con despejar la incógnita de la ecuación en la que se encuentra sola para calcular su valor. Después, este valor calculado nos sirve para sustituir en la siguiente ecuación y así obtener una nueva incógnita. Este proceso se repite hasta resolver el sistema.

Los coeficientes de las matrices [L] y [U] se obtienen directamente por igualación del producto de ambas matrices a la matriz [A]. Los resultados que se obtienen se pueden resumir en las siguientes fórmulas:

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} c_{ik} \,.\, t_{kj} \quad ,, \quad j\ \text{Í}\ i$$

$$u_{ij} = \frac{1}{c_{ii}} (a_{ij} - \sum_{k=1}^{i-1} c_{ik} \,.\, t)_{kj} \quad ,, \quad i < j$$

Este método se puede ver con mayor claridad si realizamos un sencillo programa para resolver sistemas de ecuaciones que lo utilice. La rutina de resolución que contiene, por otra parte, es la que se empleó en el programa de resolución de circuitos en continua.

```
Program Sistema_de_ecuaciones;

(* Este programa resuelve un sistema de ecuaciones por
   el método de descomposición L - U.  El programa
   principal realiza la entrada y salida de datos.  El
```

```
   procedimiento Resuelve_Sistema es el encargado de
   realizar el cálculo.  *)

(*=====================================================*)
(*=====  RESOLUCION DE UN SISTEMA DE ECUACIONES =====*)
(*=====================================================*)

Const

  Orden_Maximo = 20;
  (* Orden máximo permitido al sistema*)

Type
   Sistema = array[1..Orden_Maximo,1..Orden_Maximo]
                of real;
  (* Tipo de datos para la Matriz del sistema *)

   Vector  = array[1..Orden_Maximo] of real;
   (* Tipo de datos para los vectores de incógnitas,
      terminos independientes... *)


(* Procedimiento para resolver un sistema de
   ecuaciones mediante factorización   L - U
   Parámetros:
     - A    : Matriz con los coeficientes del sistema.
              En ella se devuelve el resultado de la
              factorización.
     - B    : Vector de términos independientes.
     - Orden: Orden del sistema.
     - X    : Vector de soluciones *)

Procedure Resuelve_Sistema(var A: Sistema;
          var B: vector; Orden: Integer;var X:Vector);

var
  i,j,k  : integer;
  Y      : Vector;
  (* Solución intermedia de LY = B *)

Begin

(* Marcha directa. Descomposición A = LU *)

  For j:=1 to Orden do
    Begin
      for i:=j to Orden do
        begin              (* Coeficientes matriz L *)
          for k:=1 to j-1 do A[i,j]:=A[i,j]
```

```
                                          -A[i,k]*A[k,j];
          end;
        for i:=j+1 to Orden do
          begin
            (* Coeficientes matriz U *)
            for k:=1 to j-1 do A[j,i]:=A[j,i]
                                          -A[j,k]*A[k,i];
            A[j,i] := A[j,i] / A[j,j];
          end;
      end;

(* Marcha inversa. Solución de LY = b *)

  For i:=1 to Orden do
    begin
      Y[i]:=B[i];
      for k:=1 to i-1 do Y[i]:=Y[i] - A[i,k]*Y[k];
      Y[i]:=Y[i] / A[i,i];
    end;

(* Marcha inversa. Solución de UX = Y *)

  For i:=Orden downto 1 do
    begin
      X[i]:=Y[i];
      for k:=Orden downto i+1 do X[i]:=X[i]
                                           - A[i,k]*X[k];
    end;
end;

(*====================================================*)
(*====FIN RESOLUCION DE UN SISTEMA DE ECUACIONES====*)
(*====================================================*)

(* Variables globales del programa *)

Var
   A: Sistema;                    (* Matriz del sistema *)
   B,                        (* Términos independientes *)
   X: Vector;                             (* Resultados *)
   N: integer;                    (* Orden del sistema *)
   i,j: integer;                (* Variables auxiliares *)


BEGIN

  WriteLn('RESOLUCION DE SISTEMAS DE ECUACIONES');
  WriteLn('------------------------------------');
  WriteLn;
```

```
Write('Orden del sistema......>');
ReadLn(N);

WriteLn;
WriteLn('Introduzca los coeficientes del sistema:');

For i:=1 to N do
  For j:=1 to N do
    begin
      Write('A[',i,',',j,'].................>');
      ReadLn(A[i,j]);
    end;

WriteLn;
WriteLn('Introduzca los términos independientes:');

For i:=1 to N do
  begin
    Write('B[',i,']...................>');
    ReadLn(B[i]);
  end;

Resuelve_Sistema(A,B,N,X);

WriteLn;
WriteLn('Descomposición del sistema:');

WriteLn;
WriteLn('Matriz L:');

for i:=1 to N do
  For j:=1 to i do
    begin
      WriteLn('L[',i,',',j,'] = ',A[i,j]);
    end;

WriteLn;
WriteLn('Matriz U:');

for i:=1 to N do
  For j:=i+1 to N do
    begin
      WriteLn('U[',i,',',j,'] = ',A[i,j]);
    end;

WriteLn;
WriteLn('Solución del sistema:');
```

```
    For i:=1 to N do
      WriteLn('X[',i,'] = ',X[i]);

  end.
```

Consideremos ahora el caso, muy frecuente en análisis de circuitos, de que tengamos un sistema de ecuaciones complejas, esto es, en el que las variables y sus coeficientes son números complejos. Tenemos dos procedimientos para atacarlo:

- Descomponer el sistema en otro de doble orden, igualando en cada ecuación por separado parte real y parte imaginaria de ambos miembros.
- Resolver el sistema de igual manera que el real, pero operando con números complejos.

Esta última solución es la que se ha llevado a cabo en el programa que a continuación vamos a analizar, y cuya rutina principal se ha empleado al analizar los métodos de resolución de circuitos en alterna.

Puesto que en Pascal estándar no existen las variables de tipo complejo, se ha definido un tipo de datos para sustituirlas, que consiste en una matriz de dos números reales. Al mismo tiempo se han realizado los procedimientos para ejecutar las cuatro operaciones básicas entre números complejos y el paso de rectangulares a polares. De este modo, la misma estructura del programa de resolución para sistemas reales sirve aquí, sin más que sustituir:

- Los elementos de las matrices de reales, por otros del nuevo tipo definido.
- Las operaciones entre dichos elementos: De las normales en Pascal a las implementadas en los nuevos procedimientos programados.

```
Program Sistema_de_ecuaciones_Complejas;

(* Este programa resuelve un sistema de ecuaciones
   complejas por el método de descomposición L - U. El
   programa principal realiza la entrada y salida de
   datos.  El procedimiento Resuelve_Sistema es el
   encargado de realizar el cálculo.  *)

(*==============================================*)
(*=RESOLUCION DE UN SISTEMA DE ECUACIONES COMPLEJAS=*)
(*==============================================*)

Const
```

```
   Orden_Maximo = 20;
                    (* Orden máximo permitido al sistema*)

 Type
   Complejo= array[1..2] of real;
                                   (* Variable compleja *)
   Sistema = array[1..Orden_Maximo,1..Orden_Maximo]
                  of Complejo;
        (* Tipo de datos para la Matriz del sistema *)

   Vector  = array[1..Orden_Maximo] of complejo;
     (* Tipo de datos para los vectores de incógnitas,
                          terminos independientes... *)

 Procedure R_P(Var C:Complejo);
                          (* Rectangulares a polares *)
 var
   R: real;
 begin
   R:=Sqrt(Sqr(C[1])+Sqr(C[2]));
   if Abs(C[1]) < 1E-10
     then
       begin
         if C[2]>0
           then C[2]:=Pi/2
           else C[2]:=-Pi/2;
       end
     else
       begin
         if C[1]>0
           then C[2]:=ArcTan(C[2]/C[1])
           else C[2]:=ArcTan(C[2]/C[1]) + Pi;
       end;
   C[1]:=R;
 end;

 Procedure IguC(C1: Complejo;var Cr:Complejo);
                                 (* Iguala complejos *)
 var
   i:integer;
Begin
  For i:=1 to 2 do Cr[i]:=C1[i];
end;

Procedure SumC(C1,C2: Complejo;var Cr:Complejo);
                                (* Suma de complejos *)
var
  i:integer;
```

```
Begin
  For i:=1 to 2 do Cr[i]:=C1[i]+C2[i];
end;

Procedure ResC(C1,C2: Complejo;var Cr:Complejo);
                              (* Resta de complejos *)
var
  i:integer;
Begin
  For i:=1 to 2 do Cr[i]:=C1[i]-C2[i];
end;

Procedure MulC(C1,C2: Complejo;var Cr:Complejo);
                           (* Producto de complejos *)
var
  i:integer;
Begin
  Cr[1]:=C1[1]*C2[1]-C1[2]*C2[2];
  Cr[2]:=C1[1]*C2[2]+C1[2]*C2[1];
end;

Procedure DivC(C1,C2: Complejo;var Cr:Complejo);
                           (* División de complejos *)
var
  i:integer;
  D: real;
Begin
  D:=sqr(C2[1])+Sqr(C2[2]);
  Cr[1]:=(C1[1]*C2[1]+C1[2]*C2[2])/D;
  Cr[2]:=(-C1[1]*C2[2]+C1[2]*C2[1])/D;
end;


(* Procedimiento para resolver un sistema de
   ecuaciones mediante factorización L - U
   Parámetros:
     - A    : Matriz con los coeficientes del sistema.
              En ella se devuelve el
              resultado de la factorización.
     - B    : Vector de términos independientes.
     - Orden: Orden del sistema.
     - X    : Vector de soluciones *)

Procedure Resuelve_Sistema(var A: Sistema;
      var B: vector; Orden: Integer;var X:Vector);

var
 i,j,k  : integer;
```

```
  Y       : Vector;
                   (* Solución intermedia de LY = B *)
  C       : Complejo;

Begin

         (* Marcha directa. Descomposición A = LU *)

  For j:=1 to Orden do
    Begin
      for i:=j to Orden do
        begin                (* Coeficientes matriz L *)
          for k:=1 to j-1 do
            begin
              MulC(A[i,k],A[k,j],C);
              ResC(A[i,j],C,A[i,j]);
            end;
        end;
      for i:=j+1 to Orden do
        begin                (* Coeficientes matriz U *)
          for k:=1 to j-1 do
            begin
              MulC(A[j,k],A[k,i],C);
              ResC(A[j,i],C,A[j,i]);
            end;
          DivC(A[j,i],A[j,j],A[j,i]);
        end;
    end;

(* Marcha inversa. Solución de LY = b *)

  For i:=1 to Orden do
    begin
      IguC(B[i],Y[i]);
      for k:=1 to i-1 do
        begin
          MulC(A[i,k],Y[k],C);
          ResC(Y[i],C,Y[i]);
        end;
      DivC(Y[i],A[i,i],Y[i]);
    end;

(* Marcha inversa. Solución de UX = Y *)

  For i:=Orden downto 1 do
    begin
      IguC(Y[i],X[i]);
      for k:=Orden downto i+1 do
        begin
```

```
          MulC(A[i,k],X[k],C);
          ResC(X[i],C,X[i]);
        end;
    end;
end;

(*=======================================================*)
(*====FIN RESOLUCION DE UN SISTEMA DE ECUACIONES====*)
(*=======================================================*)

(* Variables globales del programa *)

Var
   A: Sistema;                    (* Matriz del sistema *)
   B,                        (* Términos independientes *)
   X: Vector;                            (* Resultados *)
   N: integer;                     (* Orden del sistema *)
   i,j: integer;               (* Variables auxiliares *)

Procedure ReadLnC(var C: Complejo);
                                    (* Lectura complejo *)

begin
  Write('Parte Real: ');
  Read(C[1]);
  Write('  Parte Imaginaria: ');
  ReadLn(C[2]);
end;


BEGIN

  WriteLn('SISTEMAS DE ECUACIONES COMPLEJAS');
  WriteLn('--------------------------------');
  WriteLn;

  Write('Orden del sistema.....>');
  ReadLn(N);

  WriteLn;
  WriteLn('Introduzca los coeficientes del sistema:');

  For i:=1 to N do
    For j:=1 to N do
      begin
        Write('A[',i,',',j,']................>');
        ReadLnC(A[i,j]);
      end;
```

```
  WriteLn;
  WriteLn('Introduzca los términos independientes:');

  For i:=1 to N do
    begin
      Write('B[',i,']....................>');
      ReadLnC(B[i]);
    end;

  Resuelve_Sistema(A,B,N,X);
  WriteLn;
  WriteLn('Descomposición del sistema:');

  WriteLn;
  WriteLn('Matriz L:');

  for i:=1 to N do
    For j:=1 to i do
      begin
        WriteLn('L[',i,',',j,'] = ',A[i,j][1],' +j ',
                                    A[i,j][2]);
      end;

  WriteLn;
  WriteLn('Matriz U:');

  for i:=1 to N do
    For j:=i+1 to N do
      begin
        WriteLn('U[',i,',',j,'] = ',A[i,j][1],' +j ',
                                    A[i,j][2]);
      end;

  WriteLn;
  WriteLn('Solución del sistema:');

  For i:=1 to N do
    WriteLn('X[',i,'] = ',X[i][1],' +j ',X[i][2]);

end.
```

NOTAS PARA ADAPTAR LOS PROGRAMAS DEL PRESENTE LIBRO A SU ORDENADOR

Como ya se expuso al comienzo del libro, todos los programas se han realizado en lenguaje Pascal. Han sido compilados mediante TURBO PASCAL en un ordenador IBM-PC.

Si usted dispone de un compilador de Pascal para su ordenador no encontrará prácticamente ningún problema en ejecutar dichos programas, ya que incorporan instrucciones que son comunes a todos los compiladores de Pascal existentes en el mercado. Unicamente puede necesitar agrupar las declaraciones de constantes, tipos y variables al comienzo del programa. Para ello:

- Busque dentro del programa todas las definiciones de constantes. Las reconocerá por ir precedidas por el identificador CONST y terminar con otra palabra reservada, que podrá ser BEGIN, TYPE, VAR o PROCEDURE. Tome, por tanto, todas estas líneas y colóquelas en el programa tras el identificador de programa que se encuentra en la primera línea.
- Haga lo mismo con las definiciones de tipos. Estas las encontrará tras la palabra reservada TYPE. Coloque dichas definiciones tras el bloque de definición de constantes antes agrupado.
- Repita este mismo procedimiento con las definiciones de **variables globales**, esto es, aquellas que no se encuentran introducidas dentro de un procedimiento.

Con esta sencilla transformación podrá usted ejecutar su programa tras compilarlo con los compiladores de Pascal más comunes del mercado.

Si, por el contrario, usted sólo posee un intérprete de BASIC, la tarea puede resultar un poco más compleja, pero en ningún caso imposible. A continuación se ofrece una serie de normas maestras para ejecutar dicha conversión:

- Cambie los nombres de las variables al máximo número de letras que permita su ordenador. Este número es normalmente 2.

• Introduzca las variables alfabéticas con el identificador adecuado al intérprete BASIC que utilice. Normalmente deberá terminarlas con un «$».

• Sentencias de asignación: Mientras en PASCAL se utiliza para asignar a una variable el símbolo «:=», en BASIC se utiliza solamente «=»:

```
         PASCAL                              BASIC

     a:=b+c                              10 LET a=b+c

                                       o simplemente

                                         10 a=b+c
```

• Defina las matrices: Las variables definidas en PASCAL como ARRAY debe definirlas en BASIC con una sentencia DIM:

```
        PASCAL                               BASIC

    VAR
    V:  ARRAY[1..100,1..10]          10 DIM V(100,10)
          of INTEGER
```

• Convierta las funciones de entrada/salida:

  • WRITE o WRITELN deberá transformarla por PRINT.
  • READ o READLN deberá transformarla por INPUT.

• Sentencias de bucles (FOR) son muy fáciles de trasladar:

```
  PASCAL BASIC

FOR i:= 1 to 20 do                 10  FOR i=1 to 20
  BEGIN                            20  sentencia
    sentencia;                     30  sentencia
    sentencia;                     40  sentencia
    sentencia;                     50  NEXT i
  END;

--------------------------------------------------------
```

```
REPEAT                          10  sentencia
  sentencia;                    20  sentencia
  sentencia;                    30  sentencia
  sentencia;                    40  IF C$<>"Q" THEN GOTO 10
UNTIL Componente='Q'

---------------------------------------------------------

WHILE Control <> 'N' do         10 IF C$ = "N" THEN GOTO 60
  begin                         20 sentencia
    sentencia;                  30 sentencia
    sentencia;                  40 sentencia
    sentencia;                  50 GOTO 10
  end;                          60 sentencia
```

• Procedimientos. Esta es la parte más delicada, ya que el **BASIC** no dispone de llamada a subrutinas con paso de parámetros. En cualquier caso, hay que dar a las variables que se van a utilizar en el procedimiento los valores para los que queremos que dicho procedimiento se ejecute.

Veámoslo de forma más clara mediante un sencillo ejemplo:

```
      PASCAL
      ------
  PROGRAM Demostracion;

  VAR
    m,n,o: integer;

  PROCEDURE Ejemplo (VAR a,b,c: INTEGER);
  VAR l: integer;
  BEGIN
    l:=b*c+3;
    IF l<0 THEN
      BEGIN
        b:=0;
        c:=0
      END;
  END;

  BEGIN
    READLN(m,n,o);
```

```
  Ejemplo(m,n,o);
  WRITELN(m,n,o);
END.

    BASIC
    -----

10 INPUT(m,n,o)
20 a=m
30 b=n
40 c=o
50 GOSUB 1000
60 m=a
70 n=b
80 o=c
90 PRINT(m,n,o)
100 STOP

1000 l=b*c+3
1010 IF l>=0 THEN GOTO 1040
1020 b=0
1030 c=0
1040 RETURN
```

Es muy importante resaltar que las variables definidas dentro de un procedimiento en PASCAL (la «l» del programa anterior, por ejemplo), son **variables locales.** Esto quiere decir que no son visibles desde el resto del programa. Por tanto:

- Su valor sólo es accesible dentro del procedimiento.
- Si existiera otra variable con igual nombre en el resto del programa, su valor no se vería alterado al asignarle un valor a la variable local en cuestión.

En BASIC no existen las variables globales. Cuide, por tanto, su uso dentro de los procedimientos. Si en el ejemplo anterior «l» fuera una variable del programa principal, su valor se perdería debido a la sentencia de asignación de la línea 1000.

A continuación se ofrece el programa de resolución de sistemas de ecuaciones convertido al BASIC como ejemplo:

```
10 REM Resolución de sistemas de ecuaciones.
20 REM Orden máximo permitido al sistema
```

```
30 LET OM = 20
40 REM Matriz del sistema
50 DIM A(OM,OM)
60 REM Vector de términos independientes.
70 DIM B(OM)
80 PRINT "RESOLUCION DE SISTEMAS DE ECUACIONES"
90 PRINT "------------------------------------"
100 PRINT
110 PRINT "Orden del sistema......>";
120 INPUT N
130 PRINT
140 PRINT "Introduzca los coeficientes del sistema:"
150 FOR I=1 TO N
160 FOR J=1 TO N
170 PRINT "A(";I;",";J;").................>";
180 INPUT A(I,J)
190 NEXT J
200 NEXT I
210 PRINT;
220 PRINT "Introduzca los términos independientes:"
230 FOR I=1 TO N
240 PRINT "B(";I;")...................>";
250 INPUT B(I)
260 NEXT I
270 GOSUB 1000
280 PRINT;
290 PRINT("Descomposición del sistema:");
300 PRINT;
310 PRINT("Matriz L:")
320 FOR I=1 TO N
330 FOR J=1 TO I
340 PRINT "L(";I;",";J;") = ";A(I,J)
350 NEXT J
360 NEXT I
370 PRINT
380 PRINT "Matriz U:"
390 FOR I=1 TO N
400 FOR J=I+1 TO N
410 PRINT "U(";I;",";J;") = ";A(I,J)
420 NEXT J
430 NEXT I
440 PRINT
450 PRINT "Solución del sistema:"
460 FOR I=1 TO N
```

```
470 PRINT "X(";I;") = ";X(I)
480 NEXT I
490 STOP
1000 REM Marcha directa. Descomposición A = LU
1010 DIM Y(N)
1020 FOR J=1 TO N
1030 FOR I=J TO N
1040 FOR K=1 TO J-1
1050 A(I,J)=A(I,J)-A(I,K)*A(K,J)
1060 NEXT K
1070 NEXT I
1080 FOR I=J+1 TO N
1090 FOR K=1 TO J-1
1100 A(J,I)=A(J,I)-A(J,K)*A(K,I)
1110 NEXT K
1120 A(J,I)=A(J,I)/A(J,J)
1130 NEXT I
1140 NEXT J
1150 REM Marcha inversa. Solución de LY = b
1160 FOR I=1 TO N
1170 Y(I)=B(I)
1180 FOR K=1 TO I-1
1190 Y(I)=Y(I)-A(I,K)*Y(K)
1200 NEXT K
1210 Y(I)=Y(I)/A(I,I)
1220 NEXT I
1230 REM  Marcha inversa. Solución de UX = Y
1240 FOR I=N TO 1 STEP -1
1250 X(I)=Y(I)
1260 FOR K=N TO I+1 STEP -1
1270 X(I)=X(I) - A(I,K)*X(K)
1280 NEXT K
1290 NEXT I
1300 RETURN
```

La simulación por ordenador en general, y la de circuitos en particular, es uno de los temas que mayor interés han despertado en los técnicos desde el comienzo de la informática hasta nuestros días. En el presente libro se estudian los métodos matemáticos que permiten representar un circuito mediante fórmulas y ecuaciones, y las herramientas necesarias para aprovechar al máximo la potencia del ordenador con vistas a automatizar el proceso de diseño.